Context-driven Clustering by Multi-class Classification in an Active Learning Framework *

Martin Godec Sabine Sternig Peter M. Roth
Horst Bischof
Institute for Computer Graphics and Vision
Graz University of Technology
{godec,sternig,pmroth,bischof}@icg.tugraz.at

Abstract

Tracking and detection of objects often require to apply complex models to cope with the large intra-class variability of the foreground as well as the background class. In this work, we reduce the complexity of a binary classification problem by a context-driven approach. The main idea is to use a hidden multi-class representation to capture multi-modalities in the data finally providing a binary classifier. We introduce virtual classes generated by a context-driven clustering, which are updated using an active learning strategy. By further using an on-line learner the classifier can easily be adapted to changing environmental conditions. Moreover, by adding additional virtual classes more complex scenarios can be handled. We demonstrate the approach for tracking as well as detection on different scenarios reaching state-of-the-art results.

1. Introduction

Object detection or single target tracking can be formulated as binary classification problems, where a discriminative classifier has to distinguish the object of interest from the background. Very often the multi-modality in the data caused by large intra-class variability arises the need for a rather complex and large classifier complicates learning, reduces the evaluation speed, and may cause overfitting. Moreover, the complexity of the two classes can vary considerably. For instance, for surveillance scenarios, the resolution is low and the object class usually can be described with a simple model, whereas the background might be cluttered and arbitrarily complex.

*This work was supported by the FFG project EVis under the FIT-IT program, the FFG project HIMONI under the COMET programme in cooperation with FTW, the FFG project SECRECT under the Austrian Security Research Programme KIRAS, and the Austrian Science Fund (FWF) under the doctoral program Confluence of Vision and Graphics W1209.

A number of approaches have been proposed where the multi-modality within the data has been described by multiple classes or multiple classifiers (*e.g.*, [3, 10, 11, 21, 22]). Babenko *et al*. [3] developed a boosting algorithm suitable for multiple pose learning, where the aim is to simultaneously split the data into groups and to train a separate classifier for each group. Another approach using multiple classifiers has been proposed by Kim and Cipolla [11], where image clustering and training of multiple boosted classifiers are performed in parallel. Torralba *et al*. [21] developed a multi-class and multi-view object detector, where features used for different views or different classes are shared. A Cluster Boosted Tree has been developed by Wu and Nevatia [22]. One classifier is used for splitting the training samples into different classes by unsupervised clustering based on image features selected by a boosting algorithm. For most of these approaches the number of classes needs to be given in advance and all of them are trained in an off-line manner. Jacobs *et al*. [10] used the mixture of experts, that learn how to divide the training cases and assign each training case to one expert.

In object detection or tracking, however, often either the object of interest or the background are changing over time. Hence, an adaptive representation would be beneficial. Therefore, the goal of this work is to introduce a classifier that automatically adapts its complexity to the complexity of the current task. This is realized by a binary classifier that is built on a multi-class representation. In particular, the multi-modality in the background is described by a number of *virtual classes*, which are generated autonomously using context information (*i.e.*, for a more complex setup more classes are generated). Furthermore, we robustly adapt the classifier to changing conditions (*e.g.*, changing illumination conditions, changing backgrounds, etc.).

In the following, we first describe the concept of virtual classes for unsupervised training an adaptive, scene specific classifier. Then, we demonstrate this approach for two dif-

ferent applications, *i.e.*, object detection and tracking, on different publicly available datasets.

2. Virtual Classes for Scene-specific Classification

Using context information can significantly reduce the complexity of classification tasks. We introduce a concept for context-driven adaption of the classifier complexity to the actual task and to changing situations. An on-line multi-class classifier is used to model the multi-modality within the data. In a first stage (initialization) bootstrapping is performed to train an initial classifier. In the second stage (evaluation), the complexity of the classifier is adapted to changing situations.

2.1. Context-driven On-line Clustering

In order to adapt the complexity of a classifier to a scene, we propose to split the object as well as the background class into a number of *virtual classes*. The crucial point is, how to find these clusters. One possibility would be to manually pre-cluster the training data requiring to manually label all samples, which is tedious and often even not possible. To avoid pre-clustering, we propose a clustering approach for dealing with this intra-class variability. In particular, we apply a classifier-based bootstrapping using an on-line multi-class classifier (*e.g.*, [12, 20]) and *virtual classes*.

A virtual class can be considered a class within a multi-class classifier, if these classes are separated into negative and positive classes. Hence, each modality can be described by one single *virtual class* finally yielding a binary classification result (*e.g.*, a sample is classified as positive if it falls into one of the positive classes).

To start the clustering, we train an initial classifier c_0 discriminating the object of interest from an arbitrary background. Then this classifier is applied to the current scene, and for samples a_i that are either misclassified by the current classifier c_0 or are very close to the decision boundary a new *virtual class* with the label y_{v++} is added. In this way, the complexity of the classifier can be adapted to the complexity of the scene, *i.e.*, for more complex scenes more virtual classes are generated. The virtual class creation is described more formally in Algorithm 1 and is illustrated in Figure 1, where the input image and the clusters created within the bootstrapping stage are shown.

Algorithm 1 Virtual Class Generation

Require: Initialized Classifier $c = c_0$
Output: Final classifier: c

```
Extract background samples A_BG
for a_i ∈ A_BG do
    y = eval(c, a_i)
    if y = y_pos then
        // Add new virtual class
        update(c, a_i, y_v++)
    else
        // Update classifier
        update(c, a_i, y)
    end if
end for
```

2.2. Active Learning

After training the initial multi-class classifier in the bootstrapping stage, this classifier is able to discriminate between object and actual background. However, this initial classifier would not be able to cope with typical occurring changing environmental conditions (*e.g.*, changing illumination conditions, changing backgrounds, etc.). Hence, on-line algorithms are required, which allow to adapt to changing scenes. To reduce the learning effort (*i.e.*, the number of required samples) we use a context-driven active learning strategy.

Active learning is a widely used strategy when dealing with labeled and unlabeled data for sampling along the decision boundary in order to (a) select a reduced set of samples arranged around an optimal decision boundary and (b) to reduce the labeling effort (*e.g.*, [5, 14, 16, 23]). An active learner can be described by the quintuple (c, s, T, L, U) [14], where c is a classifier, s is a sampling function which identifies valuable samples, T is a teacher (supervisor), L is a set of labeled data and U is a set of unlabeled data. In general, an active learning process can be described as follows. First, a classifier c is trained by the labeled samples L. Then, the sampling function s selects valuable samples u_j. For those samples the teacher T assigns a label y_j, which is used to update the classifier c.

In this work, we use context information to define the sampling function s as well as the teacher T. Since Park and Choi [16] showed that it is more effective to sample the current estimate of the decision boundary, the most informative samples are those which are misclassified by the current classifier. Hence, we define our sampling function s such that it identifies samples close to the decision boundary. In particular, we run the classifier c yielding a confidence on the current sample and identify the samples which are very close to the decision border. Those samples are then labeled by using the scene context (teacher). If the decision is close to one of the actual background classes (implies small changes in the scene), the corresponding class is updated. Otherwise (if the background has changed too much) a new class is added to the multi-class classifier.

In this work, we consider two different tasks (*i.e.*, detection and tracking), where different strategies for using the context information are required. These strategies are explained in detail in Sections 3.2 and 3.3.

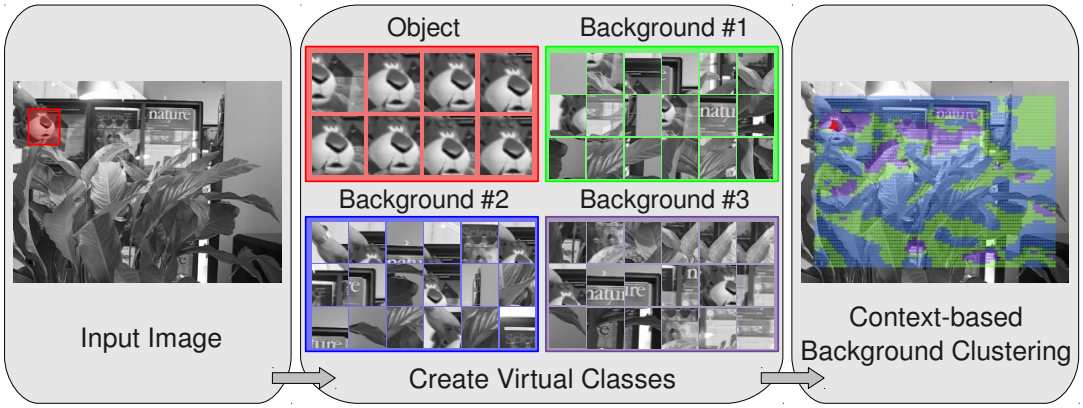

Figure 1. Generation of virtual classes: the input image is used for the bootstrapping (left), the created virtual classes (right), and an illustration of the virtual classes (middle)

3. Applications

In the following, we first describe the implementation details, *i.e.*, the used multi-class classifier with the settings used within our evaluations. Then, we discuss the two different applications, *i.e.*, object detection and object tracking and show experimental results.

3.1. Implementation Details

In general, any on-line capable multi-class classifier can be used, but in practice we use a multi-class version of on-line GradientBoost [13], related to online Multi-Class LP-Boost [19]. On-line GradientBoost combines a number of selectors f_m to one strong classifier

$$F(x) = \sum_{m=1}^{M} f_m(x). \tag{1}$$

Each selector f_m consists of a number of weak classifiers $\{f_{m,1}(x), \cdots, f_{m,N}(x)\}$ and is represented by its best weak classifier $f_{m,j}(x)$ according to the minimal error within the selector. To adapt this algorithm to a multi-class learner the weak learners $f_i(x)$ have to be able to deal with more than two classes. Thus, as weak classifiers we use on-line histograms that give a confidence rated prediction. As in [8] we use symmetric multiple logistic transformation

$$f_j(x) = \log\ P_j(x) - 1/J \sum_{k=1}^{J} \log P_k(x), \tag{2}$$

where P_j can simply be calculated in the on-line histograms. In our experiments the histograms have a size of 32 bins. As features we used Haar-like features. For the detection experiments the strong classifier consists of 50 selectors, each of it containing 20 weak classifiers. For the tracking experiments the strong classifier consists of 30 selectors, each of it containing 20 weak classifiers.

3.2. Detection with Virtual Classes

First, we demonstrate the idea of virtual classes on the task of object detection from static cameras. Initially, a classifier c_0 is trained on labeled data L. For generating the virtual classes in the bootstrapping stage we need images, which do not contain the object object of interest. For object detection from stationary cameras one can use a background model (*e.g.*, approximated median background model [15]), which can be used as an input image for the bootstrapping procedure. For false positives on the background image new *virtual classes* are created, *i.e.*, the classifier's complexity is adapted to scene's complexity. To adapt to changing situations the background model is updated all the time and used to introduce new *virtual classes* if necessary.

In order to demonstrate the performance of our approach for object detection, we evaluated it on two different publicy available standard benchmark datasets. The first dataset is *PETS 2006* [1], a pedestrian detection benchmark. The second one, the *AVSS 2007* dataset [2], is a standard benchmark for car detection. We compare our approach to two state-of-the-art object detectors, both allowing for detecting cars as well as persons. The first one is a generic detector trained off-line without scene context information, namely the deformable part model of Felzenszwalb *et al.* [7] (FS). The second approach is the classifier grid approach of Roth *et al.* [18] (CG), which is a scene specific object detector. The results are demonstrated using recall-precision curves (RPC) [2]. A detection is counted as true positive, if the overlap criteria [2] exceeds 50%, as false positive otherwise.

PETS 2006

First, we evaluated our approach on the *PETS 2006* dataset. The sequence used for evaluation consists of 308 frames (720x576 pixels).

[1] http://www.pets2006.net

[2] http://www.elec.qmul.ac.uk/staffinfo/andrea/avss2007_d.html

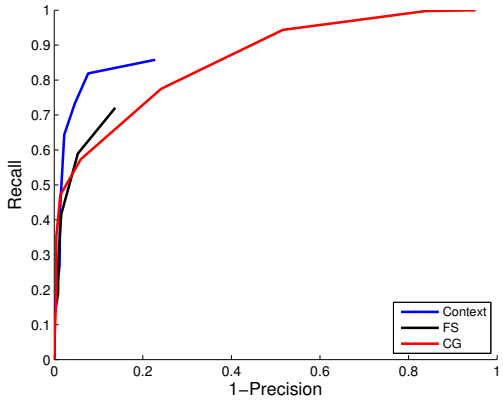

Figure 2. RPCs for the *PETS 2006 Sequence*.

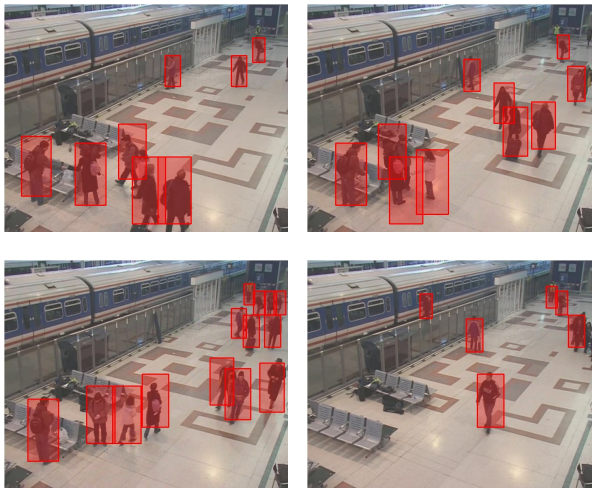

Figure 3. Illustrative detection results of our approach on the *PETS 2006 Sequence*.

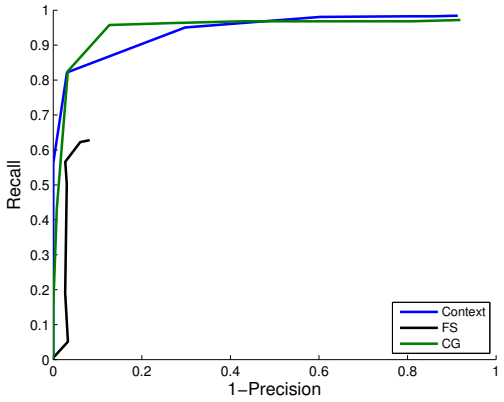

Figure 4. RPCs for the *AVSS 2007 Sequence*.

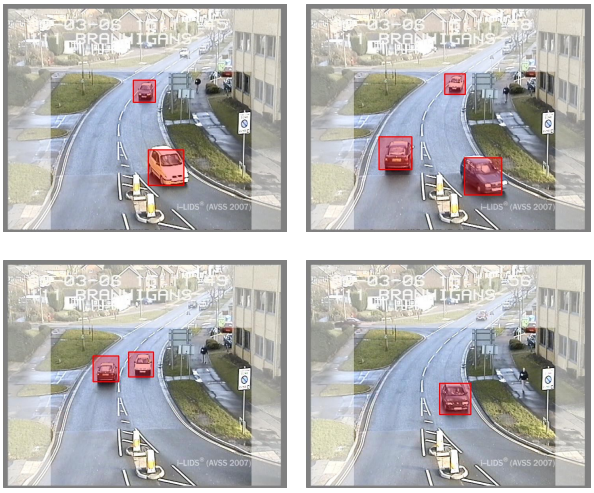

Figure 5. Illustrative detection results of our approach for the *AVSS 2007* sequence (Detection results within the fully colored region).

The Recall-Precision curves (RPC) shown in Figure 2 demonstrate that our approach outperformed both, the generic and even the scene specific approach. It can be seen that compared to these approaches, we get a higher recall still preserving the precision. Illustrative detections are shown in Figure 3.

For this sequence the number of virtual classes used to describe the multi-modality within the background class is between two and six. The variation between the number of classes is caused by the random initialization of the features within the classifier and hence the performance is not depending on the number of classes created.

AVSS 2007

In addition, we demonstrate our approach on the AVSS 2007 dataset. We evaluated on the first 500 frames (720x576 pixels) of the vehicle detection sequence AVSS_PV_Hard.

The results (see Figure 4) clearly show that we outperform the generic object detector of Felzenszwalb *et al*. [7]. Our results are comparable to the scene specific approach of Roth *et al*. [18], which has shown excellent results for the task of car detection. In addition, our approach has the advantage that instead of a large number of binary classifiers, only one multi-class classifier is used, which is much more efficient in terms of memory consumption.

For this sequence between two and five virtual classes are created during the bootstrapping stage. Since the scene is not changing over time, during run-time no further virtual classes are added to the classifier.

3.3. Tracking with Virtual Classes

Within this section, we show the performance of our approach in a *tracking-by-detection* scenario using on-line feature selection [6, 9]. Similar to the detection task, we assume that the background is only slightly changing, which holds for most of the tracking scenarios. Further, we use the context knowledge that only a single instance of the tracked

object is present in the scene at a time, which is different from the detection scenario. This allows for background updates if we know the current position of the object.

We use this context information to create and update the set of virtual classes within the multi-class classifier. At the beginning, we randomly initialize our classifier. At the first frame, we randomly select a single sample from the background and use it together with the object position to update the classifier. These two samples are enough to enable the update mechanism with virtual classes. Subsequently, we bootstrap by using every background sample already captured by a virtual background class to update this class and every false-positive sample to create a new virtual background class. During tracking, we seek for false-positives within our current scene and perform the same update strategy on the extracted patches.

For our object class, we simply perform supervised online self-learning, but semi-supervised or multiple instance updates could easily be integrated. To increase the stability of our classifier, we use two equally weighted classifiers, one is updated during runtime and one is frozen after bootstrapping, which delivers combined detections. Figure 6 compares the tracking performance with and without using virtual classes. In general, for the used tracking sequences, the amount of virtual classes was in a range of 3 to 5 during tracking. This number is directly related to the complexity of the scene and is automatically chosen by the algorithm.

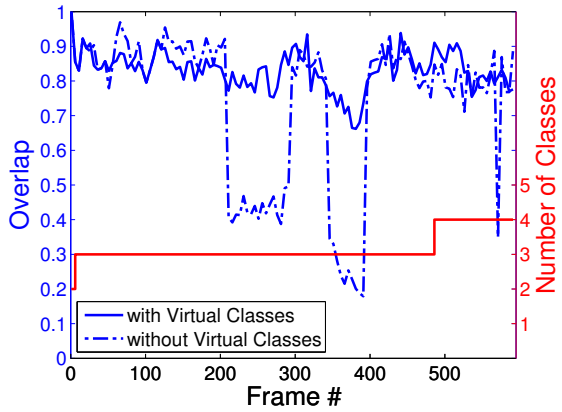

Figure 6. Comparison of tracking with and without virtual classes for the Sylvester sequence (blue solid: overlap for tracking with virtual classes; blue dashed: overlap without virtual classes - binary; red: number of virtual classes over time).

For the evaluation of our tracker we use the overlap-criterion of Agarwal [2]. This criterion is directly related to the accuracy of the detection of the classifier, in comparison to the raw distance measure between the target and background. We compute the overlap score for the entire video sequence and run each tracker 5 times, reporting the overlap score of the median run.

Table 1 lists the average overlap score for several publicly available benchmark sequences [4, 17] in comparison to other state-of-the-art tracking methods. In fact, in 4 out of 8 sequences our tracker outperforms the compared methods. For the remaining 4, it delivers state-of-the-art results close to the best method. Since we outperform both, static (Fragment-based Tracking [1] and adaptive (On-line AdaBoost [9], Multiple-Instance Tracking [4]) approaches, we show that our method is able to cope with static as well as dynamic scenarios. Figure 7 shows several illustrative samples from different tracking sequences. It is clearly visible that our tracker is able to recover if the object was occluded or the tracking result was not well aligned. With our non-optimized $C++$ implementation we achieve a frame rate of about 15 frames per second.

Sequence	CONTEXT	MIL [4]	Frag [1]	OAB [9]
Sylvester	**0.74**	0.73	**0.74**	0.62
Face 1	*0.93*	0.73	**0.94**	0.63
Face 2	**0.89**	*0.81*	0.51	*0.81*
Girl	**0.84**	0.68	*0.73*	0.57
Tiger 1	**0.65**	**0.65**	0.26	0.33
Tiger 2	*0.49*	**0.69**	0.22	0.41
David	*0.71*	**0.73**	0.52	0.39
Coke	*0.42*	**0.47**	0.10	0.25

Table 1. Average overlap score: bold-face shows the best method, while italic-font indicates the second best.

Figure 7. Illustrative tracking results on the Sylvester, Tiger1, Faceocc2, David sequences (red: our approach; blue: MIL [4]; yellow: Frag [1]; magenta: OAB [9]).

4. Conclusion

In this work, we presented an approach for automatically adapting the complexity of classifiers for object detection and tracking. In particular, we apply an on-line multi-class learner (however, finally providing a binary classifier) and introduce virtual classes to cope with the multi-modalities in the data. The approach is motivated by previous works on multi-class and multi-classifier aiming to model multi-modalities in the data. However, most of these approaches require to define the number of classes in beforehand, which is insufficient in practice since the complexity of the task is typically not known. To overcome this problem we introduce an autonomous context-driven clustering approach to generate and, on demand, to add new virtual classes and an active learning strategy to update the classes' representation. Moreover, since an on-line learner is applied this allows for adapting to changing object appearance and changing environmental condition. The approach is demonstrated for two different applications, *i.e.*, object detection and object tracking, showing that even using less complex classifiers state-of-the-art results can be obtained. Future work will concentrate on more sophisticated learning methods (*e.g.*, Multiple Instance Learner or Semi-supervised Learning) to establish a more robust foreground model and a more sophisticated strategy for enabling both, adding and removing of virtual classes.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *Proc. CVPR*, 2006. 5

[2] S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2004. 3, 5

[3] B. Babenko, P. Dollár, Z. Tu, and S. Belongie. Simultaneous learning and alignment: Multi-instance and multi-pose learning. In *Faces in Real-Life Images*, 2008. 1

[4] B. Babenko, M.-H. Yang, and S. Belongie. Visual tracking with online mulitple instance learning. In *Proc. CVPR*, 2009. 5

[5] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 1996. 2

[6] R. Collins, Y. Liu, and M. Leordeanu. On-line selection of discriminative tracking features. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2005. 4

[7] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *Proc. CVPR*, 2008. 3, 4

[8] J. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001. 3

[9] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via on-line boosting. In *Proc. BMVC*, volume I, pages 47–56, 2006. 4, 5

[10] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 1991. 1

[11] T.-K. Kim and R. Cipolla. Mcboost: Multiple classifier boosting for perceptual co-clustering of images and visual features. In *NIPS*, 2009. 1

[12] P. Laskov, C. Gehl, S. Krüger, and K.-R. Müller. Incremental support vector learning: Analysis, implementation and applications. *Journal of Machine Learning Research*, 2006. 2

[13] C. Leistner, A. Saffari A. A., P. M. Roth, and H. Bischof. On robustness of on-line boosting - a competitive study. In *Proc. On-line Learning for Computer Vision Workshop*, 2009. 3

[14] M. Li and I. K. Sethi. Confidence-based active learning. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2006. 2

[15] N. J. B. McFarlane and C. P. Schofield. Segmentation and tracking of piglets. *Machine Vision and Applications*, 1995. 3

[16] J.-H. Park and Y.-K. Choi. On-line learning for active pattern recognition. *IEEE Signal Processing Letters*, 1996. 2

[17] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *Intern. Journal of Computer Vision*, 2008. 5

[18] P. M. Roth, S. Sternig, H. Grabner, and H. Bischof. Classifier grids for robust adaptive object detection. In *Proc. CVPR*, 2009. 3, 4

[19] A. Saffari, M. Godec, T. Pock, C. Leistner, and H. Bischof. Online multi-class lpboost. In *Proc. CVPR*, 2010. 3

[20] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof. On-line random forests. In *Proc. On-line Learning for Computer Vision Workshop*, 2009. 2

[21] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2007. 1

[22] B. Wu and R. Nevatia. Cluster boosted tree classifier for multi-view, multi-pose object detection. In *Proc. ICCV*, 2007. 1

[23] R. Yan, J. Yang, and A. Hauptmann. Automatically labeling video data using multi-class active learning. In *Proc. ICCV*, 2003. 2