2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Hard Negative Classes for Multiple Object Detection

Asako Kanezaki[1], Sho Inaba[1], Yoshitaka Ushiku[1], Yuya Yamashita[1], Hiroshi Muraoka[1], Yasuo Kuniyoshi[1] and Tatsuya Harada[1]

Abstract— **We propose an efficient method to train multiple object detectors simultaneously using a large scale image dataset. The one-vs-all approach that optimizes the boundary between positive samples from a target class and negative samples from the others has been the most standard approach for object detection. However, because this approach trains each object detector independently, the scores are not balanced between object classes. The proposed method combines ideas derived from both detection and classification in order to balance the scores across all object classes. We optimized the boundary between target classes and their "hard negative" samples, just as in detection, while simultaneously balancing the detector scores across object classes, as done in multi-class classification. We evaluated the performances on multi-class object detection using a subset of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2011 dataset and showed our method outperformed a de facto standard method.**

I. INTRODUCTION

Object classification and detection are significantly in demand for intelligent autonomous systems such as robots and smart phones. The former has been taken up mainly in an image labeling task, where the system is required to output labels like "sky", "beach" or "sea" associated with an input image. The latter has been used for localizing objects corresponding to an input label like "bicycle". Both classification and detection are widely studied in several competitions such as Pascal Visual Object Classes (VOC) Challenge [6] and ImageNet Large Scale Visual Recognition Competition 2011 (ILSVRC 2011) [1], where a large scale dataset with 1.2 million training images of 1,000 object categories was used.

We claim it is often forgotten that the object detection and classification tasks are not separated but should be tackled together. In previous works, object classification has been used mainly for ranking the probabilities of the test image belonging to a set of known object classes, without considering the possibility that the test image does not belong to any of the known classes. One possible attempt is to address this issue by introducing an additional "background" class, similarly to the category structure in Caltech101 [7]. However, the assumption that there exists a single general "background" (or "negative") class does not hold when the number of target classes is large. Object detection, on the other hand, optimizes the boundary between positive samples of a target object class and negative samples. Detection, therefore, intrinsically addresses the problem of the lack of a "negative" class. However, because each detector is trained independently, the scores between target objects are not balanced relative to one another as in multi-class classification. We believe that most systems for e.g. mobile robots should be able to classify objects into multiple, properly-ranked object classes and at the time distinguish whether it belongs to a known class or not. For instance, a system should be capable of deciding that an image belongs to one class more than the other (i.e. the image is more like a motorbike than a bicycle), and at the same time judge whether the detected object belongs to a known (motorbike and bicycle) or unknown ("negative") class.

[1] Grad. School of Information Science and Technology, The University of Tokyo, Japan

We propose a novel method that trains object detection and classification simultaneously by introducing a multi-class classification algorithm into a conventional object detection framework that optimizes the boundary between each object class and a number of negative samples. The proposed method can be considered as an extension of the learning approach for Deformable Part Models (DPM) [9] to the multi-class case. Main contributions of this work are follows.

- Realized simultaneous learning of multi-class object detectors with proper score balance by alternating "hard negative" samples collection and multi-class classifiers optimization instead of one-vs-all classifiers optimization.
- Introduced as many "negative" classes as target object classes instead of a single "background" class, which enables optimization of the boundaries between each target object and its respective difficult negative samples without affecting other target objects.
- Showed our method outperformed a de facto standard method that trains each object detector independently on a subset of ILSVRC 2011 dataset (consisting of 209 categories). See Fig. 1.

Note that we evaluated performances on multi-class object detection by comparing the scores of the detected bounding boxes of all the target objects, contrary to the conventional single object detection method, which merely compares the scores of true and false positive detection of respective target objects. To achieve a better performance in this multi-class evaluation, the scores between target classes should be "balanced". In a strict sense, we should eliminate a tendency to weight scores of specific classes higher than others. To our knowledge our method is the first to balance object scores by applying a multi-class classification method to multiple object detection.

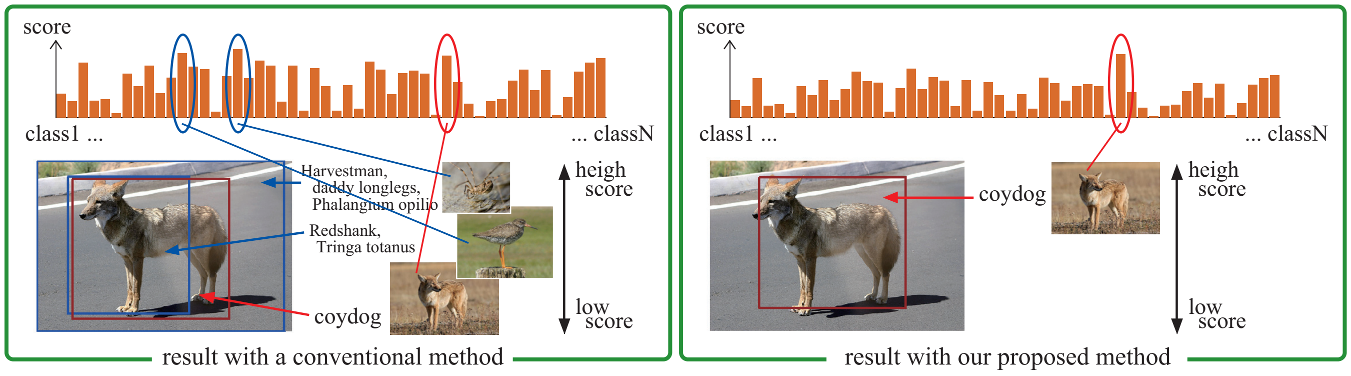

Fig. 1. Sample results with a conventional method and ours. True positive results are shown in red rectangles and false positive results with higher scores than that of true positive one are shown in blue rectangles. Although the system does detect the true object with a conventional method, it wrongly scores two false objects higher than that.

II. Related Work

Most previous works on multi-class object detection focused on utilizing "context" obtained from the co-occurrence or relative position of multiple objects in order to improve object detection performance. Some works used Conditional Random Field (CRF) to optimize labels of segments in an image [12], [13], [16], [18]. Shotton et al. [18] trained CRF parameters in this task by TextonBoost, which is a joint boosting method with shape, texture, color, location and edge cues. Rabinovich et al. [16] used a matrix of label co-occurrence counts and Google Sets which generated keywords related to input keywords, and McFee et al. [13] proposed Contextual CRF using Multiple Kernel Large Margin Nearest Neighbor (MKLMNN). Lim et al. [12] proposed a CRF method based on Shape Ordering, which represented relative position of local descriptors on principal component axes computed for each object.

Although these CRF based approaches are appealing in that they can achieve both segmentation and detection of objects, Desai et al. [5] pointed out their limitation in discriminative power due to the difficulty in building features for objects defined primarily by shape. As a possible solution, Desai et al. [5] introduced canonical relation constraints of multi-class object detection windows into the standard SVM and sliding window framework. A similar approach by Lampert et al. [10] evaluated multiple object detectors integrated with Multiple Kernel Learning and considered the existence of related objects to improve object detection performance. Wang et al. [21] also took relative positions of objects into account in a Hough voting scheme for multi-class object detection. There are also many studies on the applications of multi-class object detection. For instance, Sadeghi and Farhadi [17] achieved visual phrases like "a person riding a horse" to describe images by training combinations of multi-class objects. Li et al. [11] proposed Object Bank, which is a collection of highly discriminative image features for scene classification obtained by pooling the scores of multiple object detectors.

In these works, it was obviously critical to balance the scores across all object classes, that is, to enable score estimation in ways that take into account an object's relative likeliness to one category over another. The efficient and scalable method to achieve it, however, has not been made clear. Lampert et al. [10] trained each single-class object detector independently in a stage prior to their Multiple Kernel Learning. Desai et al. [5] optimized a vector obtained by concatenating weight vectors for descriptors and relative positions of all target classes. However, the method is not scalable to large datasets. Platt [15] trained a sigmoid function for SVM outputs as a post-processing to fit posterior probability, but it does not yet deal with the multi-class case. How can scores across multiple objects be efficiently balanced in training? Aimed for large-scale object detection tasks, Torralba et al. [19], [20] proposed the Joint Boosting which shares features among multiple object classes and reduces both computational cost and the number of required training samples. Mikolajczyk et al. [14] proposed a hierarchical tree of features based on Bayesian decision rules for efficient multi-class object detection. Although approaches which use common features or a generative model are appropriate for large-scale multi-class object detection tasks, achieving performances that are comparable to independently optimized single-class object detectors remains a challenging problem.

These days, Deformable Part Models (DPM) [9] using HOG descriptors [3] has been a de facto standard method of learning single-class object detectors. They used one-vs-all SVM considering position of object parts as latent variables, which is called Latent SVM. There are a number of works that applied DPM, including Dean et al. [4] which realized fast detection of 100,000 object classes using hashing functions. Nevertheless, it remains a challenge how to balance the scores across all object classes in order to achieve distinguishability, which is however rarely discussed in multiple object detection tasks.

III. Method

Our aim is to train discriminative multi-class object detectors efficiently. In the DPM learning framework, the training of an object detector and the re-collection of training samples are iterated one after the other. Our idea is to simply apply this framework to multiple target object classes. During the training of object detectors, we update each object detector in a way that makes its positive samples' scores higher than not only the "hard negative" samples of the target class but also the positive samples of other classes. To put it into practice, we introduce Passive Aggressive (PA) algorithm [2] instead of one-vs-all SVM. Because PA is an efficient online algorithm which can be applied to multi-class classification tasks, the computational cost of our method is comparable to the case where object detectors are trained independently. The illustration of our method is shown in Fig. 2.

The key issue on learning multi-class object detectors is how to manage huge quantities of negative samples. In a standard single-class object detection framework, "hard negative" training samples are collected into a single negative class, and then SVM is solved to separate positive and negative classes. However, in a multi-class object detection framework, "hard negative" samples vary according to object classes, which are not able to be represented by a single class model. Here, we introduce a target-specific negative class which consists of a respective target class's "hard negative" samples. We call the PA solved in this setting "multi-class backgound PA". A conventional method of learning multiple object detectors using one-vs-all SVM is illustrated in Fig. 3, and the proposed method of learning multiple object detectors using multi-class background PA is illustrated in Fig. 4. We describe the framework of learning object detectors with hard negative samples collection in Section III-A, original PA in Section III-B, and the proposed multi-class background PA in Section III-C.

A. Hard negative samples collection

To use HOG descriptors [3] and one-vs-all SVM is one of the most basic approaches of single-class object detection. Here, the bounding box regions of target objects are used as positive samples, and randomly selected sub-regions of images not including the target objects are used as negative samples. To optimize a proper boundary between the target objects and all the rest, negative samples that are close to positive samples should be selected. Therefore, it became a de facto standard method to alternate solving one-vs-all SVM and re-collecting "hard negative" samples from negative images.

Define $\boldsymbol{w}$ as a weight vector for a HOG descriptor $\Phi(\boldsymbol{x})$ extracted from a training example $\boldsymbol{x}$. A linear classifier such that SVM optimizes calculates the score of $\boldsymbol{x}$ as $f(\boldsymbol{x}) = \boldsymbol{w} \cdot \Phi(\boldsymbol{x})$. Letting $y_i \in \{1, -1\}$ be the label of $\boldsymbol{x}$, a linear SVM minimizes the following objective function:

$$L(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} \max(0, 1 - y_i f(\boldsymbol{x}_i)), \quad (1)$$

where N is the number of training samples. Then the framework of learning object detectors with re-collecting training samples is formulated as follows.

(a) update $\boldsymbol{w}$ via SVM with a fixed $f(\boldsymbol{x})$ obtained by the current $\boldsymbol{w}$.
(b) compute $f(\boldsymbol{x})$ with a fixed $\boldsymbol{w}$ by collecting new training samples $\boldsymbol{x}$ using a sliding window approach.

False positive detection windows with high scores are collected as "hard negative" samples in the latter phase (b) and are used to refine $\boldsymbol{w}$ in the next iteration of phase (a), resulting in efficient optimization of the boundary between positive samples of the target class and a large number of (or, in theory, infinite) negative samples.

This framework actually corresponds to the framework of learning DPM [9], which is a well-known state-of-the-art object detection method. Here, we briefly describe DPM. The model parameter of an object detector $\boldsymbol{w}$ consists of one root filter which is a weight vector for HOG descriptors extracted from bounding boxes of a target object class and several part filters for HOG descriptors extracted from sub regions of the bounding boxes at twice the resolution. Letting F_0 be the root filter, $F_1, \ldots, F_n$ be the part filters, (dx_i, dy_i) be deformation cost weights, and b be an offset value so that $\boldsymbol{w} \equiv (F'_0, \ldots, F'_n, d_1, \ldots, d_n, b)$, the score of a training example $\boldsymbol{x}$ is calculated by the following equation:

$$f(\boldsymbol{x}) = \max_{z \in Z(\boldsymbol{x})} \boldsymbol{w} \cdot \Phi(\boldsymbol{x}, z), \quad (2)$$

where z is a latent variable representing specification of the object configuration and $\Phi(\boldsymbol{x}, z)$ is a column vector obtained by concatenating the HOG descriptor values and deformation features extracted from a sub window. Then Equation (1) becomes a non-convex optimization problem due to Equation (2), so that phase (a) and phase (b) should be alternated to optimize $\boldsymbol{w}$ and $f(\boldsymbol{x})$.

Although our work is based on this framework of learning DPM, we use root filters with no part filter for each object detector. This is because the optimization of part filters is time consuming and not suitable for large scale multi-class object detection tasks. When we use root filters only, the model parameter for each class becomes simply $\boldsymbol{\beta} = (F'_0, b)$, and the score of an example $\boldsymbol{x}$ becomes $\boldsymbol{\beta} \cdot \Phi(\boldsymbol{x}, z) = F'_0 \cdot \phi(H, p_0) + b$. Therefore the score function of each class becomes simply $f(\boldsymbol{x}) = \boldsymbol{w} \cdot \Phi(\boldsymbol{x})$, which corresponds to the standard linear one-vs-all SVM. Notice that we still receive much benefit from the model refinement by alternating phase (a) and phase (b) collecting "hard negative" training samples. Also, the proposed approach using multi-class background PA is able to be applied to part models.

The one-vs-all SVM pipeline implemented in [8] (baseline) is summarized as follows. First, we separate the positive samples of a target class into M clusters according to the aspect ratio of their bounding boxes. Next, we break down each cluster into two clusters to separate left- and right-facing samples. We use one of the two clusters and a cluster of negative samples created by randomly choosing windows from negative images, to train the SVM and obtain a model consisting of M components. Since each component

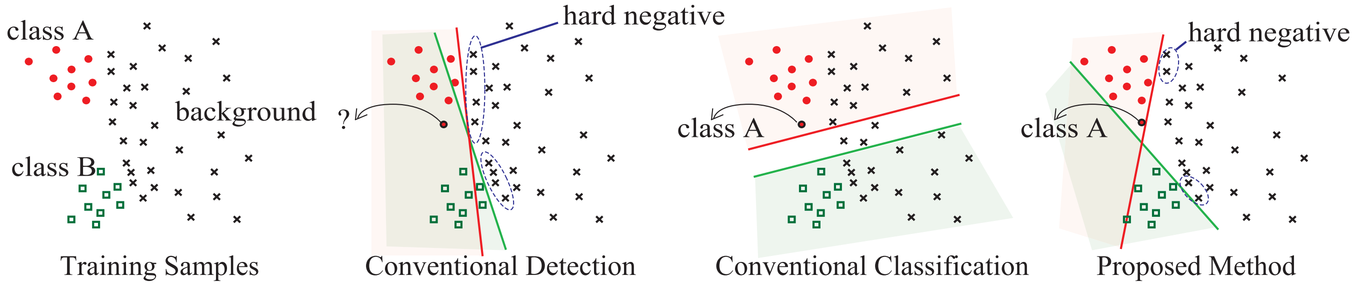

Fig. 2. Illustration of our method. Conventional object detection methods separate each class and the closest negative samples independently, failing to balance the scores across target classes. In conventional object classification methods, samples of target classes are not taken into account. Our approach optimizes the boundaries between target classes and negative samples simultaneously.

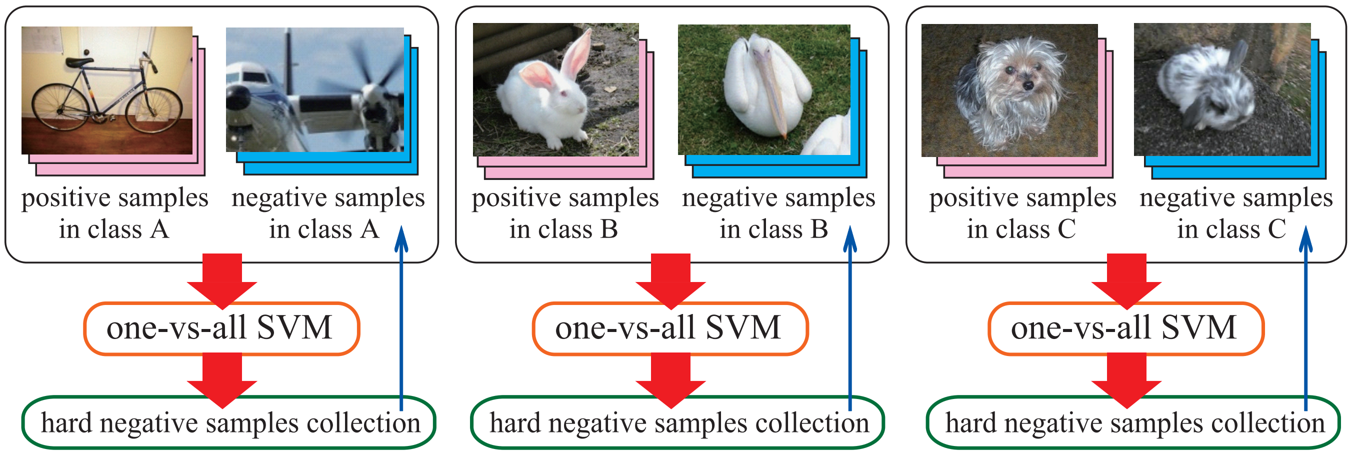
positive samples in class A
negative samples in class A
positive samples in class B
negative samples in class B
positive samples in class C
negative samples in class C
one-vs-all SVM
one-vs-all SVM
one-vs-all SVM
hard negative samples collection
hard negative samples collection
hard negative samples collection

Fig. 3. Learning of multiple object detectors using one-vs-all SVM (conventional).

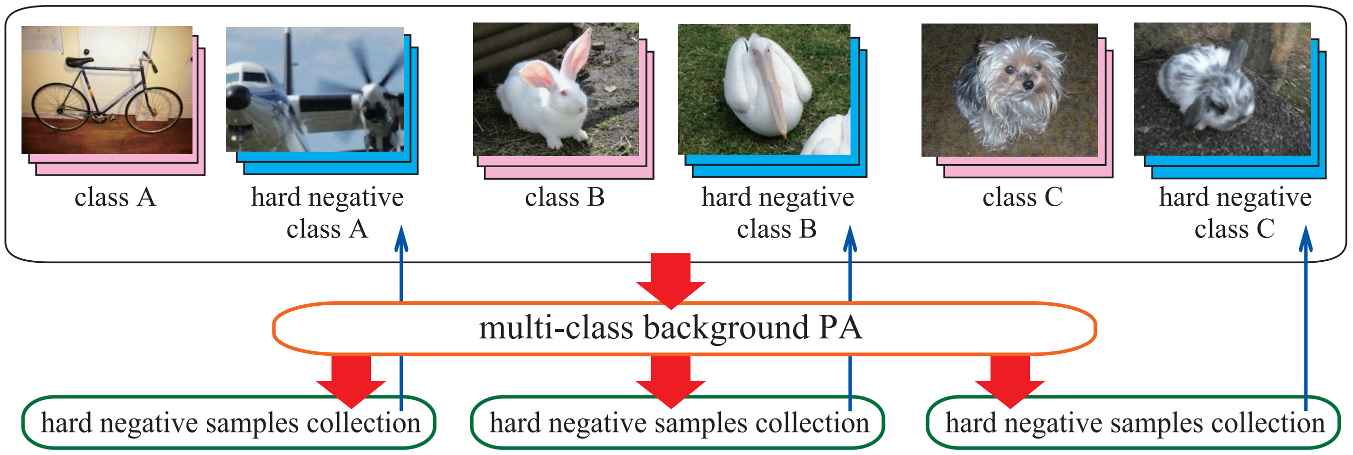
class A
hard negative class A
class B
hard negative class B
class C
hard negative class C
multi-class background PA
hard negative samples collection
hard negative samples collection
hard negative samples collection

Fig. 4. Learning of multiple object detectors using multi-class background PA (proposed).

is flipped to create its symmetric filter, we obtain $2M$ components in the initial model β_0. Then we compute $f_{\beta_0}(\boldsymbol{x})$ of training examples re-collected by sliding-window object detection using the model β_0. Here, we collect high-scored positive samples so that the intersection of their bounding boxes and those of the ground truth is larger than their union multiplied by some threshold $th_{overlap}$, and we collect hard negative samples so that their scores are higher than some threshold th_{score}. The maximum number of hard negative samples is set to be α times larger than the number of positive samples. Finally, we train SVM with the scores $f_{\beta_0}(\boldsymbol{x})$ to update the model from β_0 to β_1. These processes are iterated, updating β and $f_{\beta}(\boldsymbol{x})$ one after the other. According to [8], we set $M = 3$, $th_{overlap} = 0.7$, $th_{score} = -1$, $C = 0.02$, and $\alpha = 10$.

B. Passive Aggressive (PA)

PA [2] is an online learning method for maximum margin classification, which finds a support vector machine based on a single example while replacing the norm constraint of

SVM with a proximity constraint to the current classifier. Let us begin with a single-class classification task. The initial weight vector $\boldsymbol{w}^{(0)}$ is set to $\mathbf{0}$. Letting the weight vector in the t-th stage be $\boldsymbol{w}_t$ and the reference training sample be $< \boldsymbol{x}_t, y_t >$, $\boldsymbol{w}^{(t+1)}$ is determined as a solution of the following optimization problem:

$$\boldsymbol{w}^{(t+1)} = \arg\min_{\boldsymbol{w}} \frac{1}{2}\|\boldsymbol{w} - \boldsymbol{w}^{(t)}\|^2 \tag{3}$$

$$\text{s.t.} \max\left(0, 1 - y_t(\boldsymbol{w} \cdot \boldsymbol{x}_t)\right) = 0. \tag{4}$$

Therefore, we update $\boldsymbol{w}$ by minimizing the following objective function:

$$L(\boldsymbol{w}, \tau) = \frac{1}{2}\|\boldsymbol{w} - \boldsymbol{w}^{(t)}\|^2 + \tau(1 - y_t(\boldsymbol{w} \cdot \boldsymbol{x}_t)). \tag{5}$$

Note that we set $\boldsymbol{w} = \boldsymbol{w}^{(t)}$ and skip the update when $y_t(\boldsymbol{w} \cdot \boldsymbol{x}_t) > 1$. By solving $\bigtriangledown_{\boldsymbol{w}} L(\boldsymbol{w}, \tau) = 0$ and $\bigtriangledown_{\tau} L(\boldsymbol{w}, \tau) = 0$, $\boldsymbol{w}$ is updated as follows:

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} + \tau y_t \boldsymbol{x}_t. \tag{6}$$

The definition of τ is proposed in three different forms in [2]. In this work, we adopt the following definition (PA-I).

$$\tau = \min\left\{C, \frac{1 - y_t(\boldsymbol{w}^{(t)} \cdot \boldsymbol{x}_t)}{\|\boldsymbol{x}_t\|^2}\right\}. \tag{7}$$

C is the upper bound of τ called the "aggressiveness parameter", which is similar to the cost function in linear SVM. A large C results in faster convergence due to the aggressive updating of $\boldsymbol{w}$ but tends to lead to overfitting.

The PA algorithm is capable of being applied to a multi-class classification task. Letting the number of the target classes be K, the weight vectors of the respective classes in the t-th stage be $\boldsymbol{w}_1^{(t)}, \ldots, \boldsymbol{w}_K^{(t)}$, and the reference training sample be $< \boldsymbol{x}_t, y_t >$, the weight vectors are updated by the following equation:

$$\boldsymbol{w}_{y_t}^{(t+1)} = \boldsymbol{w}_{y_t}^{(t)} + \tau \boldsymbol{x}_t \quad \text{and} \quad \boldsymbol{w}_{s_t}^{(t+1)} = \boldsymbol{w}_{s_t}^{(t)} - \tau \boldsymbol{x}_t \tag{8}$$

$$\text{s.t.} \quad s_t = \arg\max_{s \neq y_t} \boldsymbol{w}_s^{(t)} \cdot \boldsymbol{x}_t, \tag{9}$$

where τ is computed as follows.

$$\tau = \min\left\{C, \frac{1 - \boldsymbol{w}_{y_t}^{(t)} \cdot \boldsymbol{x_t} + \boldsymbol{w}_{s_t}^{(t)} \cdot \boldsymbol{x_t}}{2\|\boldsymbol{x}_t\|^2}\right\}. \tag{10}$$

If we set $K = 2$, this formula is actually identical to that of single-class classification described above. That is to say, supposing y ($\in \{-1, 1\}$) is a class label and $\boldsymbol{w}$ is the weight vector in the single-class classification task, and supposing $\{y^1 = y, y^2 = -y\}$ are class labels and $\boldsymbol{w}_1$, $\boldsymbol{w}_2$ are the weight vectors in the two-class classification task, we obtain $\boldsymbol{w}_1 = \boldsymbol{w}/2$ and $\boldsymbol{w}_2 = -\boldsymbol{w}/2 = -\boldsymbol{w}_1$.

C. Our Model with Multiple Backgrounds

The proposed method substitutes PA in multi-class classification formulation for one-vs-all SVM in phase (a) described in Section III-A. To gain an intuitive understanding of our model, we first begin with a single-class object detection task. In this case, we apply PA with $K = 2$ while setting positive samples into a class (i.e. the target class) and setting the negative samples into another class (i.e. the background class). Then, the object detector $\boldsymbol{\beta}$ is obtained as $(F_0', b) = \boldsymbol{w}_1$. The weight vector for the background class becomes redundant because $\boldsymbol{w}_2 = -\boldsymbol{w}_1$ as described in Section III-B. Therefore, we dismiss $\boldsymbol{w}_2$ after the PA training.

Now we describe the case of multi-class object detection with $K > 1$. The simplest solution might be to set the positive samples of i-th object into i-th class and negative samples into $K + 1$-th class (i.e. the background class). However, this approach leads to poor performance, as we will see in Section IV. This is because the "background" class is optimized independently for each target class in the framework where we collect hard-negative samples which are similar to the target object. Therefore, if you put together hard-negative samples of all the target classes into a single "background" class, the boundaries between the "background" class and the target classes are confounded.

In our approach, we prepare multiple "background" classes for every target class (e.g. "non-bicycle" class and "non-chair" class and so on) and apply PA to the total $2K$ classes. Let the target class labels be $Y^+ = \{y^1, \ldots, y^K\}$ and background class labels be $Y^- = \{y'^1, \ldots, y'^K\}$. When the label of the reference training sample in the t-th stage is a positive label $y_t \in Y^+$, Equation (9) is replaced by the following equation:

$$s_t = \arg\max_{s \in \{y_t' \cup Y^+ \setminus y_t\}} \boldsymbol{w}_s^{(t)} \cdot \boldsymbol{x}_t. \tag{11}$$

When the label of the reference training sample in the t-th stage is a negative label $y_t' \in Y^-$, we set $s_t = y_t$. For a positive sample, the boundaries between its class and other target classes as well as its corresponding "background" class are updated, while for a negative sample, the boundary between its class and its corresponding target class is updated. In this way, the scores across all object classes are balanced simultaneously as the boundaries between their respective "background" classes are optimized.

Note that the case of single object detection becomes identical to the multi-class formulation with $K = 2$, and thus the weight vector of the target class in this case corresponds to one-half of that which is obtained in the single-class formulation, as described in Section III-B. Analogous to one-vs-all SVM (baseline), we set the score threshold for collecting hard-negative samples in phase (b) to $th_{score} = -1/2 = -0.5$ and the aggressiveness parameter C to 0.02.

Here, we describe how to decide the aspect ratio of the filters. First, we separate the positive samples of i-th target class into M clusters according to the aspect ratio of their bounding boxes. Next, we decide the filter sizes for respective clusters $\{w_1^i, h_1^i\}, \ldots, \{w_M^i, h_M^i\}$. Here, the filter size denotes the number of cells from which gradient histograms are computed to extract a HOG descriptor, so that the filter size is a pair of integer values such as $\{8, 10\}$. Among MN clusters in total, we bring clusters with the same filter size together into one large cluster to be used for training a single filter component. Supposing there are M' different filter sizes for MN clusters, the number of filter components becomes M'. In this work, we set M to three analogous to one-vs-all SVM (baseline).

We train the filter components independently. Suppose m-th filter component is trained using a cluster where positive samples of K_m target classes exist. Letting the number of the positive samples of the minimum target class be N_{min}, we choose N_{min} positive samples from each target class and αN_{min} negative samples from each background class randomly. Then, we update the weight vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{K_m}$ by extracting one training sample after another from the shuffled sample pool. We repeat this cycle γ times. In the last cycle, we average all the $(\alpha+1)K_m N_{min}$ weight vectors that were obtained in the updating process in order to compute the final weight vectors. We experimentally determined γ to be 5 and set α to ten in the same way as one-vs-all SVM (baseline).

IV. Results

We used a subset of the ILSVRC 2011 [1] dataset for evaluation. We selected 209 classes where the average aspect ratio of ground-truth bounding boxes is four to five out of 1,000. Because we use HOG descriptors and the dimension of a descriptor differ when the aspect ratio of a bounding box changes, we train the detector filter components with different aspect ratio independently. Note that although our method is effective particularly when the aspect ratios of training samples are similar to each other in the current case, this can be avoided by using descriptors that are independent of the bounding box aspect ratio.

We evaluated our performance with randomly chosen 10, 20, 30, 40, 50, 100, and 150 classes and with all the 209 classes, respectively. Here, we prepared five sets with different target classes and computed the average performance. In the ILSVRC 2011 dataset, there are 50 test images for each target class. We used all images of all the target classes when the number of the target classes N was 10 or 20. When N was larger than 20, on the other hand, we used a portion of them so that the total number of test images were equal to or larger than 1,000 du to limitations in computational time (34 images when $N = 30$, 25 when $N = 40$, 20 when $N = 50$, 10 when $N = 100$, 7 when $N = 150$, and 5 when $N = 209$).

To compare against the proposed method which uses PA with multiple negative classes, we also evaluated a method which uses PA with a single negative ("background") class, as well as the conventional one-vs-all SVM which trains each object detector independently. For one-vs-all SVM, we used a public source code [8] of DPM [9] modifying it to use root filters with no part filter.

For the experiment, we followed the ILSVRC 2011 protocol. Bringing out the five top ranked detection results (one for each class label) for each image, the putative detection was considered correct if the intersection of at least one bounding box with at least one ground-truth bounding box was larger than 50% of their union. We applied all the K object detectors to each test image and computed the detection results for all the classes. As a post-processing step, we applied intra-class non-maximum suppression (NMS) to the detection results of each class, or multi-class NMS to all of the detection results without distinguishing between classes.

The average correct rate versus the number of classes K is shown in Fig. 5 [1]. The proposed method reported the best performance when $K \geq 30$ in the case with intra-class NMS and when $K \geq 20$ in the case with multi-class NMS. This implies that balancing the scores across multiple object classes is especially important when K is large. The number of correctly detected images out of five when $K = 209$ for each class are shown in Fig. 8. The vertical axis represents the numbers of correctly detected images and the horizontal axis represents target object class IDs. In the top row are the results with intra-class NMS and in the bottom row are those with multi-class NMS. Contrary to one-vs-all SVM, where performance varies widely depending on classes, the performance bias is reduced in the other two cases which used PA with background class(es). On the whole, the performance with intra-class NMS is better than that with multi-class NMS. Note, however, that multi-class NMS is necessary depending on the task. For example, if you expect one exclusive class label for each detected bounding box, multi-class NMS is more appropriate than intra-class NMS. Interestingly, single-background PA out-performed one-vs-all SVM when we applied multi-class NMS, implying that balancing scores across classes via PA significantly boosts performance.

Let R denote the maximum rank of detection results considered to report correct rates. The average correct rate versus R when $K = 209$ is shown in Fig. 7. As R increases, the effect on performance due to the proposed method decreases because the task becomes more like conventional single-class object detection where the class label ranks are not evaluated (see top in Fig. 7). In the case with multi-class NMS, however, the effect is significant for all R because true positive results are affected by false positive results of other classes (see bottom in Fig. 7). As an alternate evaluation index, the per-image Average Precision (AP) which is the average AP computed by the detections of all target classes per image are shown in Fig. 6. The proposed method reported the best performance when $K \geq 20$.

Sample results with one-vs-all SVM and the proposed method are shown in Fig. 1 and Fig. 9. The results when $K = 50$ are shown in Fig. 1 and those when $K = 209$ are shown in Fig. 9. When $K = 209$, the correct object was detected with the highest score with the proposed method and not with one-vs-all SVM in 80 images (see left five columns in Fig. 9), and vice versa in 32 images (see right two columns in Fig. 9).

We used 8 core CPU (Xeon X5482 3.2 GHz) in this experiment. When each object detector was trained independently

[1]Note that intra-class NMS is meaningless in this case because we evaluated only the best scored detection for each class.

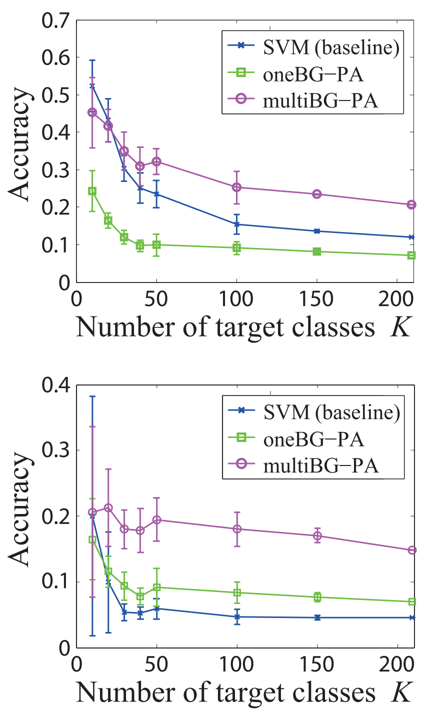

Fig. 5. Correct rate versus number of classes with intra-class NMS (top) and with multi-class NMS (bottom)

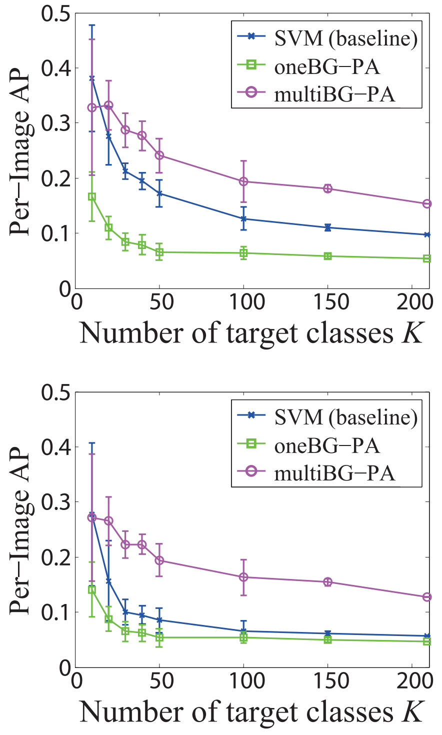

Fig. 6. Per-image AP vs number of classes with intra-class NMS (top) and with multi-class NMS (bottom)

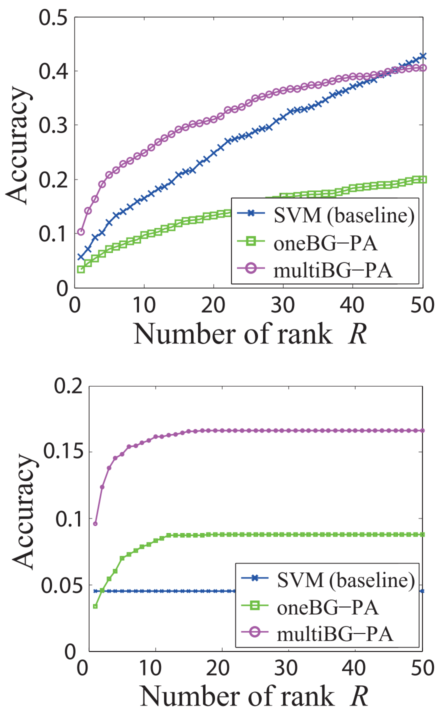

Fig. 7. Correct rate versus number of rank with intra-class NMS (top) and with multi-class NMS (bottom)

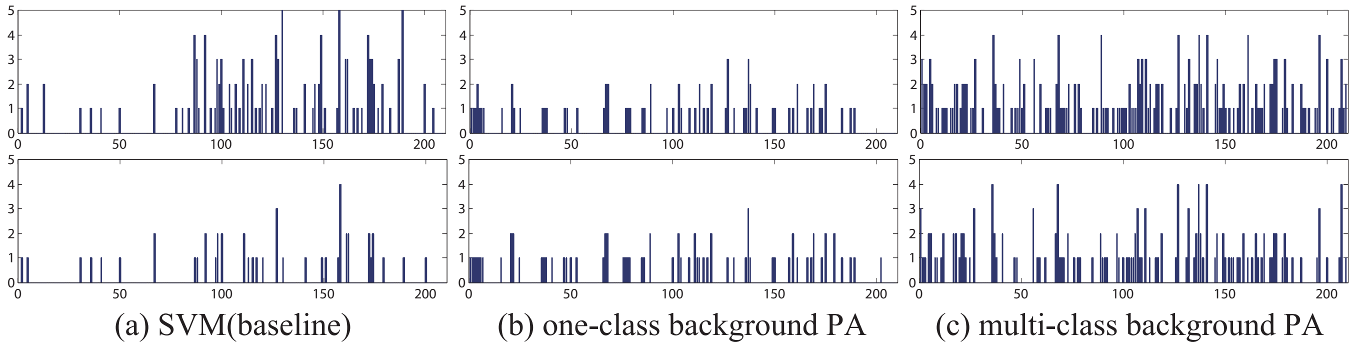

Fig. 8. Numbers of correctly detected images per each class. Results with intra-class NMS are shown in the top row and those with multi-class NMS are shown in the bottom row.

via one-vs-all SVM, the average time required to train one class across the first 50 classes (in order of class number) was 63 minutes, and the total time required to train the six slowest classes among them was 498 minutes in a single core. Therefore, the expected training time using parallel 8 cores is a minimum of 394 ($= 63 \cdot 50/8$) minutes and a maximum of 498 minutes. The time required to train 50 classes with the proposed method was 443 minutes using parallel 8 cores, which was comparable to one-vs-all SVM. In regard to execution time, it takes a couple of seconds to detect objects of one class from an image with size of 375×500 using a single core, in a similar manner as one-vs-all SVM. Most processes in one-vs-all SVM are implemented in MATLAB except for SVM training, which is implemented in C++ in voc-release4 package [8], while all processes in our proposed method are implemented in MATLAB.

V. Conclusion

We proposed a novel method for multi-class object detection, which balances the scores of multiple objects while simultaneously optimizing the boundary between each object class and "hard negative" samples collected from negative images. The proposed method is based on the strategy of re-collecting "hard negative" samples as it is used in DPM [9], which is a de facto standard method for single-class object detection. We use Passive Aggressive (PA) [2] which is applicable to multi-class classification in place of one-vs-all SVM, to train and optimize multiple object detectors simultaneously. In the experiment with a subset of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

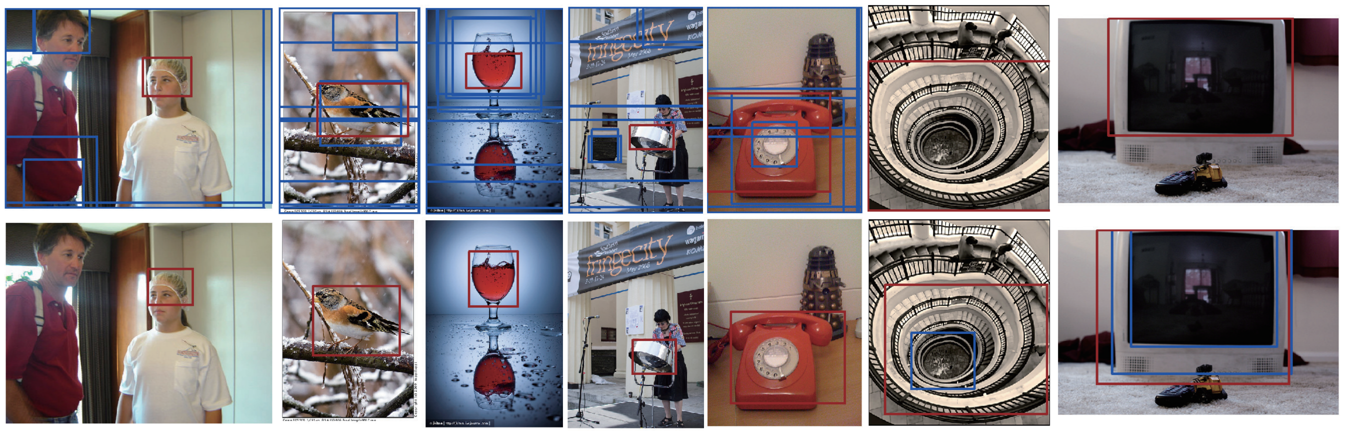

Fig. 9. Sample results with one-vs-all SVM (top row) and the proposed method (bottom row) when $K = 209$. True positive results are shown in red rectangles and false positive results with higher scores than that of true positive one are shown in blue rectangles.

2011 dataset, we showed that our method outperformed the conventional one-vs-all SVM when the number of target classes K is larger than twenty, and that our performance was twice better than that of one-vs-all SVM when $K = 209$. We showed that it is important to store "hard negative" samples for each target class into respective negative classes, rather than grouping them into a single negative class.

Currently, because we use HOG descriptors and the dimension of a descriptor differ when the aspect ratios of a bounding box changes, we train the detector filter components with different aspect ratio independently. Our future work is to use descriptors that are independent of the bounding box aspect ratio and enable simultaneous training of all filter components. To apply the proposed approach using multi-class background PA to part models is also our important future work.

References

[1] Alex Berg, Jia Deng, Sanjeev Satheesh, Hao Su, and Fei-Fei Li. IMAGENET Large Scale Visual Recognition Challenge 2011 (ILSVRC2011). `http://www.image-net.org/challenges/LSVRC/2011`.

[2] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *Machine Learning Research*, 7:551–585, 2006.

[3] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, 2005.

[4] Thomas Dean, Mark Ruzon, Mark Segal, Jon Shlens, Sudheendra Vijayanarasimhan, and Jay Yagnik. Fast, accurate detection of 100,000 object classes on a single machine. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, 2013.

[5] Chaitanya Desai, Deva Ramanan, and Charless Fowlkes. Discriminative models for multi-class object layout. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2009.

[6] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The PASCAL Visual Object Classes Challenge 2011. http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2011/index.html.

[7] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categoryies. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR) Workshop on Generative-Model Based Vision*, 2004.

[8] Pedro F. Felzenszwalb, Ross B. Girshick, and David McAllester. Discriminatively trained deformable part models, release 4. `http://people.cs.uchicago.edu/~pff/latent-release4/`.

[9] Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part based models. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 32(9), 2010.

[10] Christoph H. Lampert and Matthew B. Blaschko. A multiple kernel learning approach to joint multi-class object detection. In *Proc. DAGM symposium on Pattern Recognition*, 2008.

[11] Li-Jia Li, Hao Su, Eric P. Xing, and Li Fei-Fei. Object bank: A high-level image representation for scene classification and semantic feature sparsification. In *Proc. Neural Information Processing Systems (NIPS)*, 2010.

[12] Ser-Nam Lim, Gianfranco Doretto, and Jens Rittscher. Multi-class object layout with unsupervised image classification and object localization. In *Proc. Int. Symposium on Visual Computing*, pages 577–589, 2011.

[13] Brian McFee, Carolina Galleguillos, and Gert Lanckriet. Contextual object localization with multiple kernel nearest neighbor. *IEEE Trans. on Image Processing*, 20:570–585, 2011.

[14] Krystian Mikolajczyk, Bastian Leibe, and Bernt Schiele. Multiple object class detection with a generative model. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, 2006.

[15] John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[16] Andrew Rabinovich, Andrea Vedaldi, Carolina Galleguillos, Eric Wiewiora, and Serge Belongie. Objects in context. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2007.

[17] Mohammad Amin Sadeghi and Ali Farhadi. Recognition using visual phrases. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, 2011.

[18] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *Int. Journal of Computer Vision*, 81:2–23, 2007.

[19] Antonio Torralba, Kevin P. Murphy, and William T. Freeman. Sharing features: Efficient boosting procedures for multiclass object detection. In *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, 2004.

[20] Antonio Torralba, Kevin P. Murphy, and William T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 29:854–869, 2007.

[21] LiMin Wang, Yirui Wu, Tong Lu, and Kang Chen. Multiclass object detection by combining local appearances and context. In *Proc. ACM Multimedia*, 2011.