Indexing Ensembles of Exemplar-SVMs with Rejecting Taxonomies

Federico Becattini
University of Florence
federico.becattini@unifi.it

Lorenzo Seidenari
University of Florence
lorenzo.seidenari@unifi.it

Alberto Del Bimbo
University of Florence
alberto.delbimbo@unifi.it

***Abstract*—Ensembles of Exemplar-SVMs have been used for a wide variety of tasks, such as object detection, segmentation, label transfer and mid-level feature learning. In order to make this technique effective though a large collection of classifiers is needed, which often makes the evaluation phase prohibitive. To overcome this issue we exploit the joint distribution of exemplar classifier scores to build a taxonomy capable of indexing each Exemplar-SVM and enabling a fast evaluation of the whole ensemble. We experiment with the Pascal 2007 benchmark on the task of object detection and on a simple segmentation task, in order to verify the robustness of our indexing data structure with reference to the standard Ensemble. We also introduce a rejection strategy to discard not relevant image patches for a more efficient access to the data.**

I. Introduction

Organizing large collections of visual media is assuming an increasing importance as new content is continuously made available from social networks and scientific communities. All these new images have been used for training complex computer vision algorithms which require huge amounts of data to perform well on tasks such as image retrieval, image classification, object detection or, more broadly, scene understanding. Data therefore requires specific indexing structures in order to be handled properly and be accessed as fast as possible. To achieve this goal, various algorithms have been devised like hash functions or taxonomy learning techniques. The organization of data in such structures is guided by visual similarity and can be done exploiting images in their entirety or focusing on separate regions for each object of interest. This choice is often driven by the kind of task that we have to deal with, namely image retrieval, classification or object detection. In this paper we propose an object centric indexing strategy which adopts a metric learning approach to improve against nearest neighbor methods.

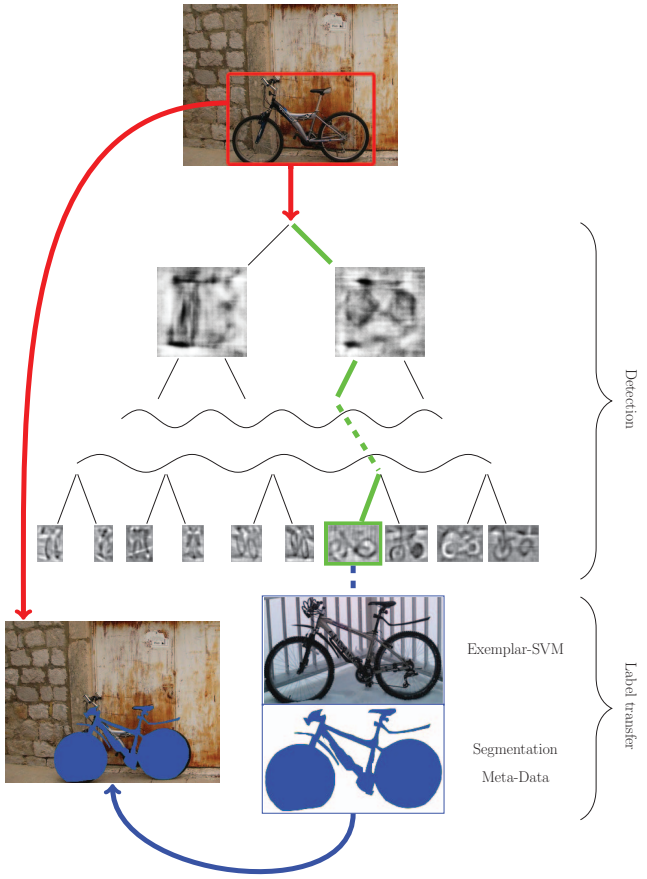

Fig. 1. Overview of our approach, exemplars of each class are stored in a learned taxonomy which is traversed to locate the best matching one enabling detection and label transfer at a logarithmic cost.

Instead of indexing image patches we organize clusters of instance specific classifiers into a taxonomy, building upon the Exemplar-SVM framework [18]. With Exemplar-SVM a classifier is trained for each available object, which represents the only positive sample opposite to a wide collection of negative patches. Thanks to this semi-parametric approach, object similarity metrics are more effective compared to nearest neighbour, since discriminative training learns highly specific templates for the object at hand. What makes the Exemplar framework even more appealing is the possibility of maintaining the properties of a nearest neighbour technique, namely the ability of establishing correspondences between train and test samples. This directly translates in the capability of transferring any available annotation assigned to train instances. In the case of visual media these annotations can vary from a simple class category identifier to fine grained labels or can be more structured information such as segmentation masks or 3D models, which can be directly associated with unlabelled data once a correspondence is established. This property has found strong interest in the computer vision

978-1-4673-8695-1/16/$31.00 2016 IEEE

community in a broad range of tasks.

The main drawback of Exemplar-SVMs, which limits the practical use of the method, is related to computational efficiency since thousands of classifiers have to be evaluated. In this paper we show that representing a learned taxonomy using a tree as indexing data structure we are able to approximate the output of a given ensemble of Exemplar-SVMs, speeding up the process of evaluation. The goal is to introduce a logarithmic factor instead of a linear one regarding the number of classifiers that need to be evaluated. Node specific thresholds are also learned for each element of the taxonomy, in order to reject test patches that are unlikely to be of any use and gain a further speed up. The choice of exploiting a taxonomy is also justified by the fact that it allows us to cluster similar Exemplars, adding implicit information about visual interclass similarity, which is useful for tasks as detection or fine grained classification.

II. Related Work

The advantages of exploiting taxonomies of visual media are twofold. On one hand computational efficiency is stressed, introducing a logarithmic dependency on the number of elements that have to be accessed. On the other hand a taxonomy will impose a topology to the data, which can be useful for classification purposes. Effectiveness and efficiency both depend of how well data is distributed and how it has been clustered together, given that the branching factor and the overall balance have an impact on both factors. With this in mind, efforts to learn an optimal taxonomy have been done.

Gao and Koller [11] learn a relaxed hierarchy in which a subset of confusing class labels can be ignored in the higher levels of the tree. This method is derived from [30] where a set of binary classifiers is organized in a tree or a DAG structure. A tree data structure where each node groups a set of category labels more and more specific as one gets closer to the leaves has been proposed in [19].

In [12] the confusion matrix of class similarities has been exploited to build a label hierarchy. A similar approach has been adopted in [3] where spectral clustering is used recursively optimizing the overall tree loss. This technique has been extended by Deng *et al.* [7] jointly learning the taxonomy structure and the classifiers used at each node through an optimization problem designed to maximize efficiency given a constraint on accuracy. Liu *et al.* [17] instead have proposed a probabilistic approach for learning the label tree parameters using maximum likelihood estimation. Similarly, random forests have been used to build fast hierarchies for classification [23] and fine-grained categorization [31].

On a related note, other general techniques for speeding up classifiers in object detection contexts have been proposed in literature. Cascade classifiers [28] have found large application in the past years, following the intuition according to which a series of week classifiers could outperform a single strong classifier. This architecture found its evolution in soft cascade [4] where the trade-off between speed and accuracy can be weighted exploring ROC surfaces. With the introduction of Felzenszwalb's DPM [10] fast evaluation techniques have been proposed exploiting Fast Fourier Transform [8], vector quantization [25] or combining various strategies to deal with different bottlenecks [24]. Optimized data structures, such as hash tables, can also be used in object detection. Dean *et al.* [6] exploit local sensitive hashing to replace the dot product operations and effectively detect up to 100.000 classes in less than 20 seconds.

In this paper we adopt taxonomies for indexing ensembles of Exemplar-SVMs (E-SVM) [18], which can benefit from such a structure to reduce the computational burden. Efforts to improve the efficiency of the Exemplar-SVM framework have been done. In [14] the problem of training thousands of classifiers has been reduced to a trivial matrix inversion with Linear Discriminant Analysis. Sadeghi and Forsyth [25] have proposed a vector quantization strategy to speed-up dot products with a lookup table, whereas Context Forests [20] have been used for predicting properties of objects exploiting their global appearance in an efficient way. New formulations of the framework have also been proposed. A joint calibration algorithm for optimizing the ensemble in its entirety is used in [21] to learn specific per-exemplar thresholds; recursive E-SVM is defined by [32], where exemplars are used as visual features encoders and in [16] three different viewpoints for interpreting E-SVM are explained. All this interest towards Exemplar-SVMs is justified by the wide range of possible label transfer applications that can be paired with object detection: segmentation [27], 3D model and viewpoint estimation [1], part level regularization [2], GPS transfer [13], scene classification [26] amongst others.

III. Complexity analysis of ESVM ensembles

To evaluate an ensemble of Exemplar-SVMs, every classifier is evaluated independently in a sliding window fashion and a set of detections is generated according to the scores of each one of them. Therefore if N windows are extracted from an image and the model contains M exemplars, the complexity is $O(NM)$. To speed this step up one must either reduce the amount of windows to be evaluated or the amount of exemplars. In the following we show how to tackle both problems. First we show how to learn a taxonomy of exemplars per class. Thanks to the fact that our learned trees are highly balanced, we are able to evaluate the exemplar ensemble with a logarithmic cost. Our proposed algorithm, in the case of a balanced binary tree has therefore a complexity of $O(N \log_2(M))$. Moreover, we show how by learning early rejection thresholds for each node of the taxonomy, the amount of windows propagated through the taxonomy can be drastically reduced obtaining an even smaller computational footprint.

IV. Learning an ESVM taxonomy

We propose to build a highly balanced tree using spectral clustering hierarchically. In a set of preliminary experiments we tested several clustering algorithms for the taxonomy

FUNCTION splitSet($\mathcal{W}$)
Data: $\mathcal{W} = \{\mathbf{w}_i \dots \mathbf{w}_M\}; S$
Result: subsets $\mathcal{W}_L, \mathcal{W}_R : \mathcal{W}_L \cap \mathcal{W}_R = \emptyset$;
$\mathcal{W} = \mathcal{W}_L \cup \mathcal{W}_R$;
subset representatives: $\hat{\mathbf{w}}_L$, $\hat{\mathbf{w}}_R$
if $|\mathcal{W}| > 1$ **then**
  $[\mathcal{W}_\mathcal{L}, \mathcal{W}_\mathcal{R}] \leftarrow$ `spectralClustering`$(\mathbf{S}, 2)$;
  $\hat{\mathbf{w}}_\mathbf{L} \leftarrow \frac{1}{|\mathcal{W}_L|} \sum_{i \in \mathcal{W}_L} \mathbf{w}_i$;
  $\hat{\mathbf{w}}_\mathbf{R} \leftarrow \frac{1}{|\mathcal{W}_R|} \sum_{i \in \mathcal{W}_R} \mathbf{w}_i$;
  splitSet($\mathcal{W}_L$);
  splitSet($\mathcal{W}_R$);
else
  leaf node reached;
end

Algorithm 1: Taxonomy Learning. We recursively apply spectral clustering obtaining a binary tree and keeping as representative for node n the mean hyperplane $\hat{\mathbf{w}}_n$.

learning, namely agglomerative clustering with various cluster aggregation criteria and hierarchical k-means.

All but the divisive spectral clustering resulted in poor taxonomies with unbalanced trees which are detrimental to performance; moreover methods based on Euclidean metrics do not work well in high-dimensional spaces, as also shown in [15], [26] due to the well known curse of dimensionality.

Spectral clustering is a technique that reformulates the problem of clustering as a graph partitioning problem, where clusters are identified through connected components. In spectral clustering the attention is focused on graph Laplacian matrices, a tool that measures the similarity of nearby vertices. In literature various formulations of the Laplacian matrix have been proposed. In this paper we follow the approach that refers to the normalized version of [22]:

$$\mathbf{L}_n = \mathbf{D}^{-1/2}\mathbf{S}\mathbf{D}^{-1/2} \tag{1}$$

where $\mathbf{S}$ is the affinity matrix and $\mathbf{D}$ is a diagonal matrix where each element $D_{ii} = \sum_{j=1}^{N} S_{ij}$.

We first define matrix $\mathbf{A}$ that captures how likely is that exemplar $\mathbf{w}_i$ fires on samples on which another exemplar $\mathbf{w}_j$ fires. This matrix represents the compatibility of exemplars. So given the matrix obtained by concatenating all the raw features $\mathbf{H} = [\mathbf{h_1} \dots \mathbf{h_N}]$ and the matrix of the respective learnt exemplar hyperplanes $\mathbf{W} = [\mathbf{w_1} \dots \mathbf{w_N}]$ our affinity matrix is defined as:

$$\mathbf{A} = \mathbf{W}^T\mathbf{H}. \tag{2}$$

Therefore element A_{ij} represents the score of exemplar $\mathbf{w}_i$ on the feature $\mathbf{h}_j$ on which exemplar $\mathbf{w}_j$ has been trained and vice versa.

Since the matrix $\mathbf{A}$ is not guaranteed to be symmetric we apply the same strategy as in [3] and define

$$\mathbf{S} = \frac{1}{2}\left(\mathbf{A}^T + \mathbf{A}\right) \tag{3}$$

that is symmetric.

Our tree is created recursively applying spectral clustering as shown in Algorithm 1. Note that for each split we set the representative of each node as $\hat{\mathbf{w}}_N = \frac{1}{|\mathcal{W}_N|}\sum_{i \in \mathcal{W}_N} \mathbf{w}_i$. An example of the representatives for the first split of the bicycle class is show in Figure 2, highlighting how the dominant views of the object, frontal and sideways, are captured. Even though the proposed approach is feature independent in our experiments we employed Histogram of Oriented Gradients features (HOG) [5], as in the original Exemplar-SVM formulation.

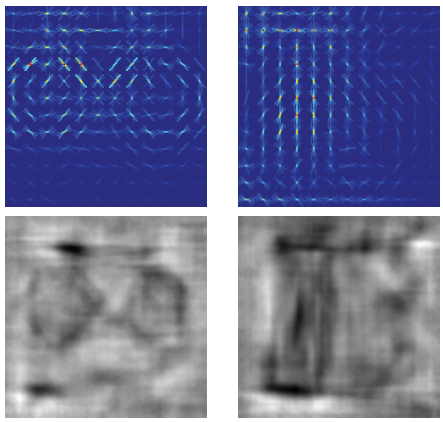

Fig. 2. HOG representation of the bicycle centroids (top) and the respective inverted HOG (bottom) [29]. It can be clearly seen that the exemplars indexed in the left child are mostly lateral views of bicycles while in the right child are frontal.

V. Accelerated ESVM evaluation

An ensemble of exemplars can be easily represented as a set $\mathcal{W}$ of exemplar hyperplanes $\mathbf{w}_i$ of the same class. Each image window feature vector $\mathbf{h}$ can be evaluated selecting the best exemplar using:

$$\arg\max_{i \in \mathcal{W}} \mathbf{w}_i^T\mathbf{h} \tag{4}$$

This requires to evaluate each window against all exemplars. Thanks to the learned taxonomy we can reduce this computation to a tree traversal. Tree traversal is performed by iteratively selecting from the current node, the child with the highest scoring representative:

$$\texttt{next\_node} = \arg\max_{i \in \{L,R\}} \hat{\mathbf{w}}_i^T\mathbf{h} \tag{5}$$

where L and R are the left and right children of the current node, respectively.

The scores $\hat{\mathbf{w}}_i^T\mathbf{h} = \frac{1}{|\mathcal{W}_N|}\sum_{i \in \mathcal{W}_N} \mathbf{w}_i^T\mathbf{h}$ represent a lower bound on the score obtained by the best exemplar present in each subtree therefore we greedily pursue the path that maximizes this bound. This of course does not guarantee to select the leaf with the actual maximum. A schematic pipeline of our system is shown in Figure 1.

A. Early rejection

For each query image we have to evaluate tens of thousands of windows, which is very expensive. Inspired by the similarities with a cascade classifier [4], we inserted in our framework a rejection threshold strategy. The aim is to discard most of the windows at the higher levels of the tree if they do not look promising. This is done learning a threshold on the output score of each classifier, which can drastically reduce the number of comparisons for each image, leading to a considerable speed-up. As can be seen in Figure 3 after a few nodes most of the background windows are discarded while windows that are propagated to the leaves are densely clustered around the object to be detected.

We propose an approach close to the philosophy of soft-cascades proposed by Bourdev and Brandt [4] to learn the threshold values. Soft-cascades are based on the usage of a rejection distribution vector $\mathbf{v} = (v_1, ...v_T)$ where $v_t \geq 0$ is the minimum fraction of objects that we are allowed to miss at the t-th stage of the cascade. In our setting, instead of a cascade with T classifiers we have a tree with M leaves, i.e. $2M-1$ classifiers. The aim is to learn a rejection threshold for each node and to do so we group together nodes at the same depth, in order to establish the rejection distribution vector. Each path down the tree is treated as a soft-cascade. For a tree of depth D we employ a rejection distribution vector $\mathbf{v}$ with D values, one for each tree level, defined as $v_d = \exp \frac{1}{2}\left(\frac{d}{D} - 1\right)$. In our implementation this function defines the percentage of windows we want to keep at each level of our tree and has the property to saturate towards 1 descending towards a leaf. With this strategy we reject most of the least promising windows in the first levels, increasing the amount of kept windows at each stage.

Thresholds t_n for each node n are learned by evaluating approximately 2M windows drawn randomly from a validation set. Considering a set of windows H that reach node n we select the value that at this node allows $v_t \cdot |H|$ windows to be retained.

VI. Experimental results

We evaluate the proposed algorithm on the Pascal VOC 2007 object detection benchmark [9], comparing the results to the baseline given by the original ESVM framework [18] and focusing on the speed-up gain in the evaluation step. We also present some label transfer results for a segmentation task, showing how the two methods are able to provide comparable high quality masks.

A. Object Detection

Object detection accuracy is important in order to establish how the taxonomy is able to approximate the capabilities of a standard ESVM ensemble. To build the hierarchical ensembles we use the pre-trained exemplars provided by [18] for each of the Pascal `trainval` objects (20 categories, 12608 exemplars). Each class is evaluated using a different taxonomy, specific to its class.

We report the results obtained using four different approaches: the standard hierarchy built through plain spectral clustering (`Tree`), an augmented version created using both exemplars and their horizontally flipped copies (`Tree-flip`) and the rejecting version of the two previous strategies (`Tree-rej` and `Tree-flip-rej`, respectively).

The flipped exemplar version of the tree reduces the cost of evaluating both windows from the original image and from its flipped copy, as commonly done to enhance detectors performance. Testing flipped windows or evaluating flipped detector models does not produce any substantial difference, but thanks to the logarithmic factor introduced by the tree, we are able to halve the number of windows to evaluate without almost any additional cost. In fact if the tree is well balanced, doubling the number of exemplars only increases tree depth by one.

To obtain the flipped version of each classifier we simply swap its components according to the orientations of the gradients in the correspondent HOG feature vector without any additional training.

After evaluating the taxonomy we rescore the best 5% of the results with the whole ensemble, in order to gather more detections. In fact the ESVM framework boosts detected bounding boxes using an exemplar co-occurrence matrix, which is less effective if we have a reduced number of detections, as with the taxonomy which associates only one detection for bounding box. The rescoring step allows us to overcome this problem by evaluating the entire ensemble only on a restricted subset of windows, which requires a negligible additional cost. A similar approach is used in [25] to mitigate the effect of the quantization approximation.

Table I summarizes the results for all of the proposed methods, along with the ESVM ensemble baseline (`ESVM`) and the Fast Template Vector Quantization [25] method applied to Exemplar-SVM. Detection accuracy is reported in terms of mean Average Precision (mAP) and the evaluation time is an $image \times exemplar$ time i.e. a cost which scales both with the number of images to evaluate and with the number of classifiers.

We used the the Exemplar-SVM framework of [18] available online to perform the evaluation of the baseline and FTVQ. All experiments have been performed on an Intel Core i7-3610QM, 4 x 2.3Ghz.

The `Tree` method obtains a mAP of 18.65, which is comparable to the ESVM ensemble baseline and requires nearly half the time to be evaluated. The use of the flipped exemplars strategy instead, at almost no additional cost in mAP, lowers by 65% the overall evaluation time since we have to evaluate only the original unflipped windows, obtaining a $2.5\times$ speed-up with respect to the ESVM baseline.

Employing rejecting thresholds we are able to reach a $4\times$ and $6\times$ speed up for the `Tree-rej` and the `Tree-rej-flip` methods, respectively. These two methods leave the mAP of the system almost unchanged with respect to the standard `Tree` algorithm, meaning that we are able to prune windows that do not contain any detectable object.

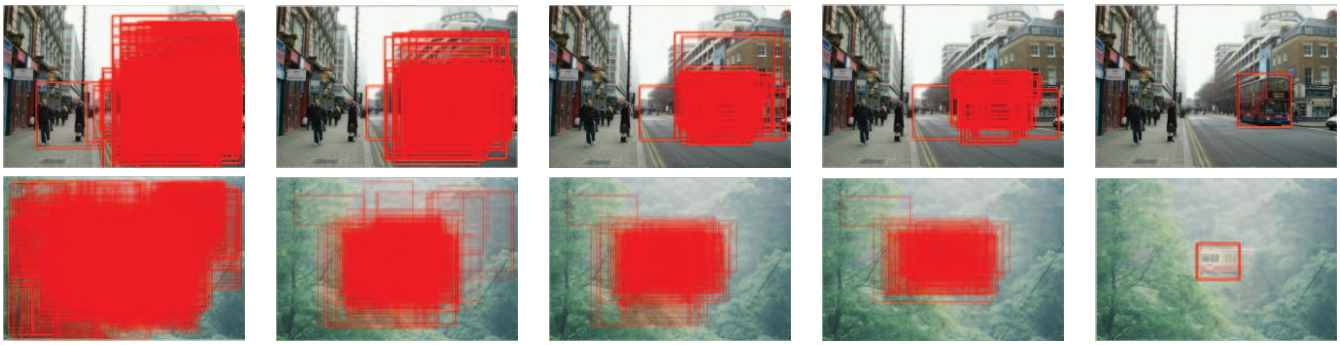

Fig. 3. Windows retained during tree traversal using rejection thresholds. The amount of windows decreases significantly from the first tree node (leftmost) to the last (rightmost).

Method	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	dining table	dog	horse	motorbike	person	potted plant	sheep	sofa	train	tv monitor	mAP	Time
ESVM [18]	**19.0**	**47.0**	**3.0**	**11.0**	9.0	39.0	**40.0**	**2.0**	6.0	15.0	**7.0**	**2.0**	**44.0**	**38.0**	**13.0**	**5.0**	**20.0**	**12.0**	**36.0**	**28.0**	**19.8**	4.6 ms
FTVQ [25]	18.0	**47.0**	**3.0**	**11.0**	9.0	39.0	**40.0**	**2.0**	6.0	15.0	**7.0**	**2.0**	**44.0**	**38.0**	**13.0**	**5.0**	**20.0**	**12.0**	**36.0**	**28.0**	19.7	1.4 ms
Tree-flip-rej	18.0	45.5	2.2	9.0	**9.4**	39.0	37.4	1.6	**6.2**	13.5	6.1	1.2	41.9	37.8	8.8	3.0	17.1	10.5	31.1	27.2	18.3	**0.8** ms
Tree-rej	13.1	46.6	1.9	10.0	8.0	39.7	37.8	1.2	5.9	**15.5**	5.1	1.6	41.6	36.0	9.7	3.4	16.8	11.2	35.0	26.8	18.4	1.2 ms
Tree-flip	17.6	45.5	2.3	9.2	7.7	39.0	38.0	1.5	**6.2**	13.6	6.1	1.2	42.0	37.6	10.6	3.0	17.2	10.8	30.4	27.1	18.3	1.8 ms
Tree	13.0	46.4	2.1	10.5	7.2	**40.0**	38.5	1.3	5.9	**15.5**	**7.0**	1.5	42.0	37.5	11.7	3.3	17.7	10.9	34.0	27.0	18.7	2.7 ms

TABLE I

RESULTS ON THE PASCAL VOC 2007 DATASET. COMPARISON OF OUR METHOD WITH THE EXEMPLAR-SVM BASELINE [18]. THE RESULTS OBTAINED USING THE FAST TEMPLATE VECTOR QUANTIZATION [25] ARE ALSO GIVEN AS A TERM OF COMPARISON. THE FOUR VARIANTS OF OUR APPROACH (STANDARD TREE, TREE WITH FLIP AUGMENTATION, REJECTING TREE, REJECTING TREE WITH FLIP AUGMENTATION) ARE REPORTED, SHOWING DETECTION ACCURACY AND TIMINGS. RUNNING TIMES REFER TO THE MEAN TIME REQUIRED FOR EVALUATING EACH EXEMPLAR ON AN IMAGE. NOTE THAT OUR METHOD IS COMBINABLE WITH THE FTVQ APPROACH IN ORDER TO GAIN A BIGGER SPEED-UP.

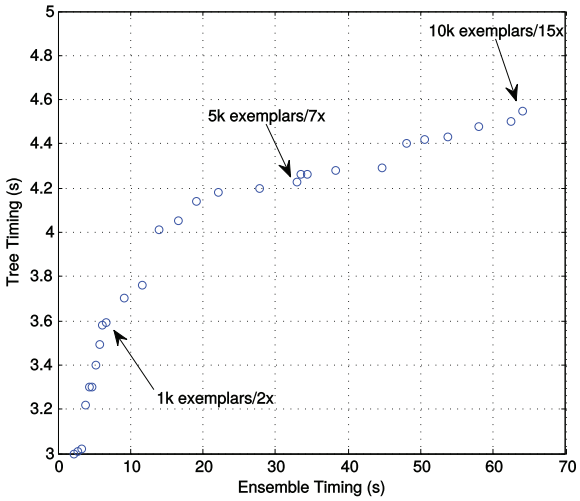

Fig. 4. Scatter plot of timings varying the number of exemplars. A logarithmic trend is observed in relation to the number of exemplars.

To give a more accurate timing analysis in Figure 4 we show a scatter plot of the image evaluation timings of the `Tree` method, compared against the baseline. Each point in the plot compares the timings obtained with the same number of exemplars. The plot shows a clear logarithmic relationship between the timings of the ensemble and our approach. It is important to note that the speed-up increases as more models are used. For example we are able to obtain a 15× speed-up using 10k exemplars. In the experiments reported in Table I this speed-up is not appreciable since most of the classes in Pascal have a small number of exemplars (200-300 on average) and we are averaging timings on all classes.

B. Label Transfer Segmentation

Along with object detection accuracy we evaluated the label transfer capabilities of the system performing a segmentation experiment. In order to do so, we manually annotated the Pascal VOC *Bus* class (229 exemplars and 213 test objects for a total of 442 buses) with segmentation masks. Using the results obtained during the detection step we were able to segment test objects by weighting the masks of all the exemplars that generated a detection according to detection confidence and by applying a threshold to remove noise.

The segmentation masks have been evaluated for both the *Bus* class and the *Background* class, measuring a pixel-wise accuracy $Acc = \frac{tp}{tp+fp+fn}$ as specified in the Pascal guidelines for segmentation. Using the hierarchical structure we obtained a 48.07% accuracy for the *Bus* class and 76.30% for the *Background* against a baseline of 49.59% and 77.14%, respectively. Some qualitative results are reported in Figure 5 where the segmentation masks generated by our method and by the baseline are compared. These results show that with our indexing strategy we are still able to retrieve objects which maintain a good alignment with test objects, leaving the label transfer capabilities of the framework almost unaffected.

Fig. 5. Segmentation masks produced by Exemplar-SVM label transfer. Green masks in the upper row are produced using the standard ensemble [18], red masks in the lower row are generated using our hierarchical ensemble.

VII. Conclusion

Given the wide range of different application scenarios, Exemplar-SVMs have received a large amount of interest in the past years. The real applicability of this technique is however still limited due to the linear dependency of the computation time in the number of examples. In this work we have proposed a technique to overcome this drawback by building an indexing structure to access data at a logarithmic cost and just a small loss in detection performance. Moreover we have shown how our method does very little harm to label transfer performance.

Combining all of our speed-up techniques our method runs faster than [25] which to the best of our knowledge is the fastest exemplar SVM evaluation method. Finally the proposed strategy can be combined with several recent improvements in the object detection literature in terms of speed-up such as vector quantization [25], [24] or FFT template convolution [8].

References

[1] M. Aubry, D. Maturana, A. A. Efros, B. C. Russel, and J.Sivic. Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models. In *Proc. of CVPR*, 2014. 2

[2] Y. Aytar and A. Zisserman. Enhancing exemplar svms using part level transfer regularization. In *BMVC*, pages 1–11, 2012. 2

[3] S. Bengio, J. Weston, and D. Grangier. Label embedding trees for large multi-class tasks. In *Proc. of NIPS*, 2010. 2, 3

[4] L. Bourdev and J. Brandt. Robust object detection via soft cascade. In *Proc. of CVPR*, 2005. 2, 4

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. of CVPR*, pages 886–893, 2005. 3

[6] T. Dean, M. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, and J. Yagnik. Fast, accurate detection of 100,000 object classes on a single machine. In *Proc. of CVPR*, Washington, DC, USA, 2013. 2

[7] J. Deng, S. Satheesh, A. C. Berg, , and L. Fei-Fei. Fast and balanced: Efficient label tree learning for large scale object recognition. In *Proc. of NIPS*, 2011. 2

[8] C. Dubout and F. Fleuret. Exact acceleration of linear object detectors. In *Proc. of ECCV*, pages 301–311, 2012. 2, 6

[9] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. 4

[10] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Pattern Analysis and Machine Intelligence*, 32(9), Sept. 2010. 2

[11] T. Gao and D. Koller. Discriminative learning of relaxed hierarchy for large-scale visual recognition. In *Proc. of ICCV*, 2011. 2

[12] G. Griffin and P. Perona. Learning and using taxonomies for fast visual categorization. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008. 2

[13] P. Gronat, G. Obozinski, J. Sivic, and T. Pajdla. Learning and calibrating per-location classifiers for visual place recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 907–914, 2013. 2

[14] B. Hariharan, J. Malik, and D. Ramanan. Discriminative decorrelation for clustering and classification. In *Computer Vision–ECCV 2012*, pages 459–472. Springer, 2012. 2

[15] A. Jain, A. Gupta, M. Rodriguez, and L. S. Davis. Representing videos using mid-level discriminative patches. In *Proc. of CVPR*, 2013. 3

[16] T. Kobayashi. Three viewpoints toward exemplar svm. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2765–2773, 2015. 2

[17] B. Liu, F. Sadeghi, M. Tappen, O. Shamir, and C. Liu. Probabilistic label trees for efficient large scale image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 843–850, 2013. 2

[18] T. Malisiewicz, A. Gupta, and A. A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *Proc. of ICCV*, 2011. 1, 2, 4, 5, 6

[19] M. Marszałek and C. Schmid. Constructing category hierarchies for visual recognition. In *Proc. of ECCV*. Springer, 2008. 2

[20] D. Modolo, A. Vezhnevets, and V. Ferrari. Context forest for object class detection. 2

[21] D. Modolo, A. Vezhnevets, O. Russakovsky, and V. Ferrari. Joint calibration of ensemble of exemplar svms. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 3955–3963. IEEE, 2015. 2

[22] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Proc. of NIPS*, 2002. 3

[23] M. Ristin, M. Guillaumin, J. Gall, and L. Gool. Incremental learning of ncm forests for large-scale image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3654–3661, 2014. 2

[24] M. Sadeghi and D. Forsyth. 30hz object detection with dpm v5. In *Proc. of ECCV*, volume 8689 of *Lecture Notes in Computer Science*, pages 65–79. Springer International Publishing, 2014. 2, 6

[25] M. A. Sadeghi and D. Forsyth. Fast template evaluation with vector quantization. In *Proc. of NIPS*, pages 2949–2957, 2013. 2, 4, 5, 6

[26] S. Singh, A. Gupta, and A. A. Efros. Unsupervised discovery of mid-level discriminative patches. In *Proc. of ECCV*, 2012. 2, 3

[27] J. Tighe and S. Lazebnik. Finding things: Image parsing with regions and per-exemplar detectors. In *Proc. of CVPR*, 2013. 2

[28] P. Viola and M. J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, May 2004. 2

[29] C. Vondrick, A. Khosla, T. Malisiewicz, and A. Torralba. Hoggles: Visualizing object detection features. In *Proc. of ICCV*, 2013. 3

[30] M. C. Y. Chen and J. Ghosh. Integrating support vector machines in a hierarchical output space decomposition framework. In *Proc. of IGARSS*, 2004. 2

[31] B. Yao, A. Khosla, and L. Fei-Fei. Combining randomization and discrimination for fine-grained image categorization. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1577–1584. IEEE, 2011. 2

[32] J. Zepeda and P. Perez. Exemplar svms as visual feature encoders. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3052–3060, 2015. 2