Multi-component Models for Object Detection

Chunhui Gu[1], Pablo Arbeláez[2], Yuanqing Lin[3], Kai Yu[4], and Jitendra Malik[2]

[1] Google Inc., Mountain View, CA, USA
chunhui@google.com
[2] UC Berkeley, Berkeley, CA, USA
{arbelaez,malik}@eecs.berkeley.edu
[3] NEC Labs America, Cupertino, CA, USA
ylin@sv.nec-labs.com
[4] Baidu Inc., Beijing, China
yukai@baidu.com

Abstract. In this paper, we propose a multi-component approach for object detection. Rather than attempting to represent an object category with a monolithic model, or pre-defining a reduced set of aspects, we form visual clusters from the data that are tight in appearance and configuration spaces. We train individual classifiers for each component, and then learn a second classifier that operates at the category level by aggregating responses from multiple components. In order to reduce computation cost during detection, we adopt the idea of object window selection, and our segmentation-based selection mechanism produces fewer than 500 windows per image while preserving high object recall. When compared to the leading methods in the challenging VOC PASCAL 2010 dataset, our multi-component approach obtains highly competitive results. Furthermore, unlike monolithic detection methods, our approach allows the transfer of finer-grained semantic information from the components, such as keypoint location and segmentation masks.

1 Introduction

Consider the object in the center of Figure 1. Although its appearance is very different from any of the surrounding instances, they all belong to the same semantic category "aeroplane". The main causes of intra-class variations in recognition are pose and viewpoint changes, as well as the presence of visually heterogeneous subcategories. For instance, aeroplanes look quite different from side to 45-degree views, and their appearance also changes significantly among the three main subcategories: wide-body passenger jets, fighter jets and propeller aeroplanes. We refer to such visual clusters as *components*.

In this paper, we propose an approach that models each component independently, which we show is easier and more accurate than attempting to characterize all components in a monolithic model. Another significant advantage of our approach over monolithic models is that it enables tasks that are finer-grained than bounding box prediction. Objects in the same component are tight in configuration space, and thus inference on the object keypoint locations and

A. Fitzgibbon et al. (Eds.): ECCV 2012, Part IV, LNCS 7575, pp. 445–458, 2012.

Fig. 1. One key challenge of object categorization is intra-class variations induced by pose changes, subcategories, etc. Since objects belonging to the same category form clusters in the appearance and configuration spaces, it is natural to construct individual models for each cluster and combine them later at the category level. We refer to such a cluster as *component*.

segmentation masks becomes feasible. The keypoints and mask of an object can be predicted from those of its most likely component.

While monolithic models are still common in the literature [7,21], there have been several influential approaches modeling multiple components of objects [10,12,5,17]. Nevertheless, each of these methods has it own limitations. Felzenszwalb et al.[10] learn global and part components jointly, but the number of components is pre-defined and not inferred from data. Gu and Ren[12] focus on modeling only viewpoint variations of objects and ignore other sources of intra-class variations such as subcategories. Bourdev et al.[5] use keypoints to align objects but their poselet models typically characterize parts rather than global objects. Malisiewicz et al.[17] is most similar to our work. However, that approach uses only one positive instance for training, which significantly reduces the generalization capacity of each component model and therefore compromises categorization performance. Last but not least, all these methods use expensive multi-scale window scanning for object detection, which sets a limit on the number of components as well as on the ability to apply more sophisticated features and more powerful classifiers for better accuracy.

To reduce the computation cost during detection, we adopt the popular idea of object candidate selection [11,1,8,14,21], but implement our own bounding box generation scheme based on bottom-up segmentation. Our scheme produces fewer than 500 bounding boxes per image on the VOC2010 dataset, drastically reducing the search space when compared to exhaustive multiscale window scanning, while maintaining high recall of objects over all 20 categories. Furthermore, since this scheme does not rely on category-specific knowledge, the number of candidate windows is independent of the number of categories and thereby scalable to large data.

Overall, this paper presents three distinct contributions that, when combined, provide competitive performance on the PASCAL detection challenge while

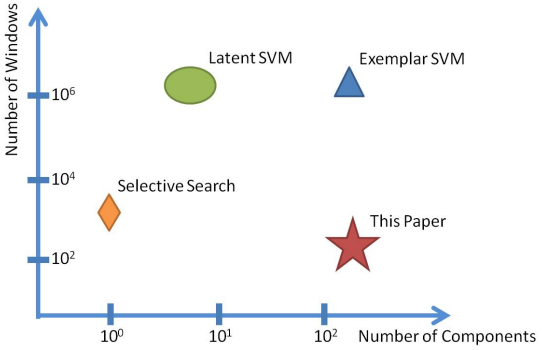

Fig. 2. Comparison of our approach with related previous methods (Latent SVM by [10], Exemplar SVM by [17], and Selective Search by [21]) in 2D space where the two axes represent the number of components and the number of window candidates per image. Our approach is distinct from others by combining multi-component models and selective window candidates.

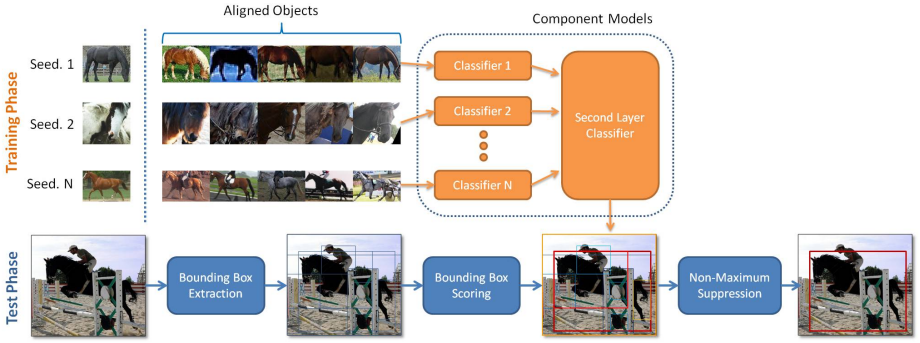

Fig. 3. An overview of our detection pipeline. In the training phase (top row), we pick a seed object from training data, and align the rest of the objects with the seed based on keypoint and mask annotations. The top aligned objects are then used as positive set for learning a single-component classifier. Given N seeds, we have N such classifiers. A second-layer classifier takes the outputs of these component classifiers as input, and produces a final category-level classification score. In the test phase (bottom row), we generate a small number of bounding boxes using bottom-up segmentation cues to avoid exhaustive window scanning. Each candidate box is then scored by our learned two-layer classifiers. Finally, a non-maximum suppression is applied to generate final detection results.

enabling finer-grained tasks than bounding box prediction: (1) global and generic multi-component models characterizing intra-class variation; (2) a category level classifier aggregating responses from multiple components; (3) a simple yet effective algorithm allowing prediction of object keypoint locations and masks. Figure 2 depicts various detection methods in a 2D plot that characterizes the

number of components of an object model and the number of scanned windows per image, respectively. This work is, to the best of our knowledge, the first one addressing the combination of multi-component models and selective window candidates.

Figure 3 gives an overview of our detection framework. In the training phase, a two-layer model is learned to capture and aggregate the components of an object category from data. Each first-layer model is a binary classifier trained with a seed and a list of aligned objects. In the detection phase, a small number of candidate bounding boxes are generated for each image using our selection scheme. After scoring these boxes with our two-layer model, a non-maximum suppression is applied to produce final detection results.

The rest of the paper is organized as follows. In Section 2, we describe our bounding box generation scheme. Sections 3 and 4 show how to find and train a component model. Section 5 describes the mechanism of combining component model outputs into a final category-level classifier. We discuss the experiments in Section 6 and conclude in Section 7.

2 Bounding Box Generation

Exhaustive window scanning is computationally expensive when the number of components scales up to hundreds. Therefore, a bounding box selection scheme is necessary to prune out windows in an early stage that do not contain any object. In this paper, we start by applying the segmentation algorithm of [3] which produces a pool of overlaid segments over scales for an input image. Since the algorithm uses *gPb* contour signals as input which recovers almost full recall of object boundaries at a low threshold, the output segments encode the sizes and shapes of objects in the input image quite precisely.

Next, each segment in the pool proposes a bounding box which is the smallest rectangle containing it. This proposal gives us the same number of bounding box candidates as the number of segments in the pool. Some candidates are identical, even though their original segments are different. After duplicate removal, we end up with on average fewer than 500 candidate boxes per image on the PASCAL VOC 2010 training set. Figure 4 shows for each category the recall of objects

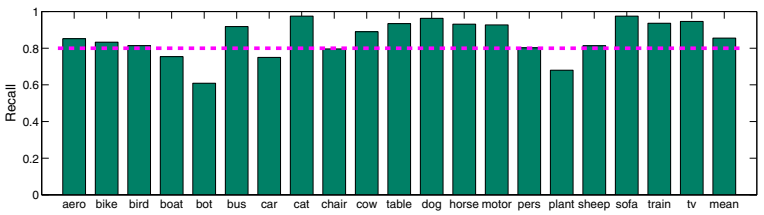

Fig. 4. Percentage of actual objects found by our bounding box generation on all 20 PASCAL VOC categories. We obtain a recall of 80% or higher among 16/20 categories.

whose ground truth bounding boxes overlap more than 50% with at least one of our proposed boxes. In 16/20 categories, we have a recall rate of 80% or higher.

Our object candidate selection scheme proposes bounding boxes efficiently and category-independently, thus avoiding unnecessary redundancies both in the image space and across categories. It provides a huge saving of computation during detection.

3 Finding Components

Clustering of all training data of an object category seems to be a natural strategy for finding the components of that category, since objects belonging to the same component, by definition, have smaller appearance and configuration distances to each other. However, in practice, this strategy does not work well. One main reason is that the components that are less common are very difficult to discover because they are easily absorbed by the components of common poses or dominant subcategories. Furthermore, objects within a cluster are, in many cases, not tight enough to build a robust component model because no global alignment is enforced among them during clustering.

Therefore, we apply a different two-step strategy to construct each of our components:

- A "seed" object is picked from the training data which characterizes a component.
- The rest of the training objects of the category are aligned to the seed through a global transformation using the keypoint and mask annotations. The top aligned objects constitute the positive set of the component.

We use the annotation data from [6] for keypoints and masks of training objects. These annotations are crowdsourced from Amazon Mechanic Turk. For each category, 10 to 20 semantically meaningful keypoints (e.g. head, tail, wing tips, wing bases, and stabilizer tip for aeroplane) are marked on the objects. Invisible keypoints are not labeled and thus excluded in the global transform step. In addition, object masks are also labeled using a polygon approximation.

With these additional annotations in hand, we recover a similarity transformation between two objects and use it to align one object to the other. Precisely, let I and J be two objects that need to be aligned, and p_I, p_J, M_I, M_J be their keypoints and masks, respectively. The transformation matrix $\mathbb{T}$ has a close-form solution when the objective is to minimize the Procrustes distance between two sets of keypoints p_I and p_J. The quality of the alignment can be measured by the following distance function:

$$d_{quality} = (1 - \lambda) \cdot d_{procrustes} + \lambda \cdot (1 - d_{overlap})$$

where $d_{procrustes} = \sum_{i \in I} (\mathbb{T}(p_J^i) - p_I^i)^2$ is the Procrustes distance, and

$$d_{overlap} = \frac{Area(M_I \cap \mathbb{T}(M_J))}{Area(M_I \cup \mathbb{T}(M_J))}$$

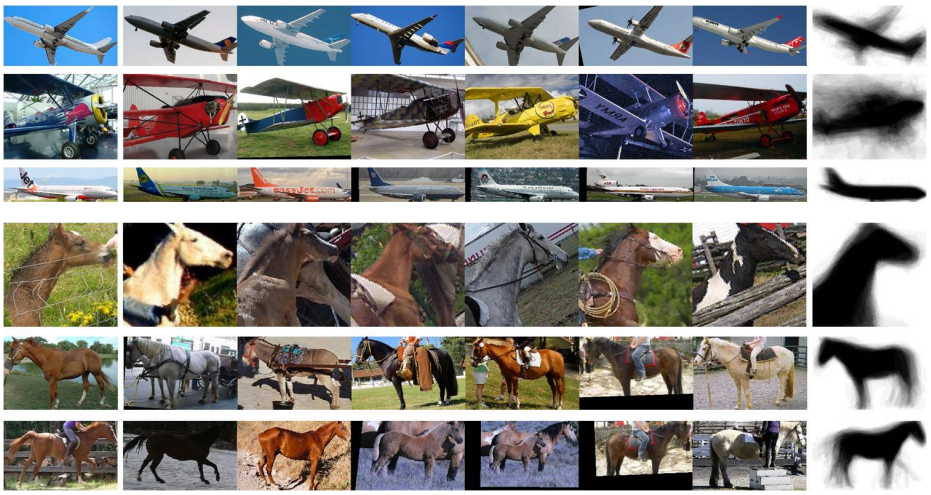

Fig. 5. Visualization of some of our components for aeroplane (top) and horse (bottom) categories. Each row corresponds to a component. The left-most images are seeds; the middle six images are top aligned objects for each seed; and the right-most images are averaged mask of top 32 aligned objects. Note that, due to our global alignment strategy, objects within each component are tight in configuration space.

is the intersection-over-union score between the seed mask and the transformed object mask. The parameter λ controls the relative weight between the two terms. In our experiments, we observe that even $\lambda = 1$(only the mask scores count) gives reasonable alignment results. Finally, we sort all aligned objects for a given seed based on the quality distances defined above, and pick top 32 as positive instances of the component model. In the PASCAL VOC 2010 data, 32 is an empirical choice that is small enough to exclude degraded object alignment for most components of categories, and big enough to make the model generalizable.

With one component model set up, we can easily extend this two-step scheme and construct a set of component models by picking multiple distinct seeds from the training data. Again, our strategy prevents less-common components from being absorbed by common ones. Objects within each component are tight in configuration space after global alignment which enables us to train strong classifiers. Figure 5 shows our alignment results on the aeroplane and horse categories with three components each. Each component model is a binary classifier and we will describe the training framework in the next section.

4 Training Components

Each of our component models is a binary classifier characterizing a particular category component. Given the set of aligned objects obtained from the

previous section, we conduct the following two-step strategy to complete component training process:

- We construct a negative set from all bounding box candidates extracted from negative training images for each component.
- We learn an Intersection Kernel Support Vector Machine (IKSVM[16,22]) based on the positive and negative sets for each component, and the model is bootstrapped once by data-mining hard negatives. The SVM scores are then converted to probabilities through a sigmoid function whose parameters are also learned from data.

We use four types of features to describe our bounding boxes, three of them using a spatial pyramid pooling, and the last one using object-centric pooling. Each component model learns all the feature types separately. Our second layer classifier, which we will describe in the next section, aggregates the outputs of all components of all feature types.

4.1 Spatial Pyramid of Vector Quantized SIFT

We implement a spatial pyramid of vector quantized SIFT[13] in the standard way: interest points are extracted from an image in a grid basis. A set of three-scale opponent-SIFT[20] descriptors are computed and concatenated to form a vector at each interest point. These vectors are then quantized to codewords based on a class-specific codebook. The size of our codebook is 400. Next, we divide each bounding box into 2×2 cells and count the frequencies of the codewords within each cell where interest points lie. The final descriptor of the bounding box is the concatenated histograms of codewords within each cell.

4.2 Spatial Pyramid of Poselet Activations

The implementation of this feature is similar to that of the vector quantized SIFT, except that we replace the SIFT-based codewords by poselet activations. Poselets[4] have been shown powerful in describing shapes for characteristic parts of objects. Compared to SIFT features, poselet activations are more sparse, informative, and discriminative. Each poselet fires only twice per image on average on the VOC dataset, and provides both strength and rectangular support of the activation. Each poselet model is trained with a highly discriminative classifier. We use pre-trained models of [5]. A typical number of poselets per category is 100 to 200. We apply a "soft count" strategy to aggregate poselet activations within an image cell. Denote $H(C, w)$ as the bin value for poselet index i in the histogram of image cell C.

$$H(C, i) = \sum_{a \in A} S(a) \times \frac{Area(B(a) \cap B(C))}{Area(B(a))} \times \mathbf{1}(I(a) = i)$$

where $I(a)$, $S(a)$ and $B(a)$ are the index, strength, and support of the activation a, and $B(C)$ is the support of C. Note that we soft count the strength of an

activation by the fraction of overlap between the support of the activation and the image cell. It proves essential to smooth out the spatial quantization noise caused by the activation location with respect to image cells.

4.3 Spatial Pyramid of Local Coordinate Coding with HOG

The work in [15] demonstrates a state-of-the-art image classification system using spatial pyramid of local coordinate coding (LCC) on local descriptors. Here we implement a similar feature representation. We sample HOG descriptors[7] on a dense grid with step size of four pixels. Then each HOG descriptor is coded using local coordinate coding [23] with codebook size of 8192. Finally, the coding results are max-pooled with a spatial pyramid representation of 1×1 and 3×3 cells.

4.4 Object-Centric Spatial Pooling

The work in [19] demonstrates that object-centric spatial pooling (OCP) is more effective than traditional spatial pyramid matching(SPM) based pooling for bounding box classification. Given a bounding box, OCP pools separately the foreground and background regions to form a bounding box representation. In contrast to traditional SPM that only performs pooling on foreground, OCP includes pooling on background and is able to provide more accurate localization. This is because the learned object detector with OCP will prevent the leakage of parts of a object into background.

For each candidate bounding box, we generate its feature representation using object-centric pooling. The way to generate the feature is the same as in Section 4.3 except that the SPM pooling was replaced by OCP.

5 Second-Layer Classifier and Non-Maximum Suppression

We leverage our learned component classifiers for the task of categorical prediction through learning a second-layer classifier, taking all component outputs of a bounding box into an input vector, and outputting a single probability score for that box during detection. Same as the design of component classifiers, our second-layer is a binary Intersectional Kernel SVM for each category, enabling the importance of components to be reflected by its learned weights. In order to avoid overfitting, we use a validation set, separated from the training set that we use for learning component classifiers, for cross-validating the parameter choices of our second-layer classifier.

We notice that the choices of positive and negative bounding boxes have a large impact on the detection performance during second-layer training. Table 1 shows various design choices on the aeroplane category and compares their detection performance. On the positive data side, enriching the positive set through adding near-duplicates of positive boxes improves the overall performance. Since

Table 1. Design choices of second-layer classifier on the aeroplane category of VOC 2010 val dataset using the spatial pyramid of poselet activation features. We notice that including only hard boxes for negative data and having up to 4 near-duplicates as positive data both improve the detection performance (mean Average Precision).

Negative Set		Positive Set		
All Boxes	Hard Boxes	No Duplicate	up to 2 Dup	up to 4 Dup
.302	.420	.382	.407	.420

Table 2. Detection results on VOC 2010 **val**: In order to better understand the power of each individual feature and their combinations, we run control experiments on the validation set of VOC 2010 and compare performance using different types of features. Note that features contribute differently to different categories, and feature combination is essential for improved performance. S,P,L,O stands for VQ'ed SIFT, VQ'ed poselet, LCC HOG, and Object-centric features, respectively.

	aer	bik	bir	boa	bot	bus	car	cat	cha	cow	din	dog	hor	mot	per	pot	she	sof	tra	tvm
S	.373	.348	.092	.061	.171	.411	.297	.251	.043	.133	.093	.152	.314	.322	.177	.061	.217	.169	.178	.300
P	.420	.361	.167	.108	.171	.585	.321	.266	.117	.218	.193	.272	.325	.433	.255	.150	.308	.260	.350	.422
L	.352	.443	.139	.094	.194	.537	.375	.356	.137	.292	.191	.273	.378	.490	.194	.170	.317	.230	.361	.372
O	.529	.294	.108	.103	.081	.469	.248	.451	.036	.102	.186	.284	.201	.386	.220	.048	.193	.198	.320	.374
SP	.454	.415	.174	.112	.216	.548	.356	.335	.111	.217	.179	.252	.392	.430	.289	.138	.337	.277	.348	.428
SPL	.457	.469	.215	.113	.242	.602	.421	.397	.153	.352	.242	.350	.466	.532	.289	.183	.414	.310	.412	.478
SPLO	.568	.434	.248	.164	.234	.635	.384	.568	.134	.298	.302	.430	.425	.514	.332	.163	.411	.381	.472	.482

some positive objects may correspond to a large number of bounding boxes, we apply a multiple-instance-learning[2] framework to automatically pick best set of near-duplicate boxes for those objects. On the negative data side, we choose to only include boxes where at least one component classifier fires (we call them "hard boxes") in the negative set. This choice enables component selection, as bad components usually fire everywhere and can be easily identified by this reduced set.

After all proposed bounding boxes are scored by our two layer classifiers, we apply non-maximum suppression on these boxes to generate detection results. The bounding boxes are sorted by their detection scores, and we greedily select the highest scoring ones while removing those that are sufficiently covered by a previously selected bounding box. We use 30% as the coverage threshold based on cross-validation results.

6 Experiments

6.1 Object Detection on PASCAL VOC

We use the standard PASCAL VOC [9] platform to benchmark our detection framework. Each of our component models is trained on VOC 2010 train data, and evaluated on 2010 val. We use all but no more than 500 objects from training data as seed objects. In addition, each seed and its aligned objects are mirrored

Table 3. Detection Results on VOC 2010 **test**: This table compares our full system with other leading approaches on VOC 2010 test data. For each category, the winner is shown in **bold** font and the runner-up in *italics*. The performance of our approach is highly competitive.

	aer	bik	bir	boa	bot	bus	car	cat	cha	cow	din	dog	hor	mot	per	pot	she	sof	tra	tvm
[5]	.332	.519	.085	.082	*.348*	.390	*.488*	.222	-	.206	-	.185	.482	.441	**.485**	.091	.280	.130	.225	.330
[10]	.524	*.543*	.130	.156	**.351**	.542	**.491**	.318	.155	.262	.135	.215	.454	.516	*.475*	.091	.351	.194	.466	.380
[21]	**.582**	.419	**.192**	.140	.143	.448	.367	**.488**	.129	.281	**.287.394**	.441	.525	.258	**.141.388.342**	.431	**.426**			
NLPR	.533	**.553.192.210**	.300	*.544*	.467	.412	**.200.315**	.207	.303	*.486*	*.553*	.465	.102	.344	.265	**.503**	.403			
[24]	*.542*	.485	.157	*.192*	.292	**.555**	.435	.417	.169	.285	.267	.309	.483	.550	.417	.097	*.358*	*.308*	.472	*.408*
NUS	.491	.524	.178	.120	.306	.535	.328	.373	*.177*	*.306*	*.277*	.295	**.519.563**	.442	.096	.148	.279	*.495*	.384	
Ours	.537	.429	.181	.165	.235	.481	.421	*.454*	.067	.234	*.277*	*.352*	.407	.490	.320	*.116*	.346	.287	.433	.392

Table 4. Detection Results on VOC 2007 **test**: This table compares the results on VOC 2007 test set between our multi-component(MC) model and the monolithic(MN) model using the same feature set and training data. Note the improvement of multi-component model over monolithic model on almost every category. In addition, our model also outperforms [17] that trains each component model using only one positive instance.

	aer	bik	bir	boa	bot	bus	car	cat	cha	cow	din	dog	hor	mot	per	pot	she	sof	tra	tvm	avg
MN	.248	.268	.059	.109	.092	.381	.375	.228	.097	.163	**.236**	.147	.252	.260	.177	.104	.197	.211	.210	.358	.209
MC	**.334**	.370	**.150.150.226.431.493.328**	.115	**.358**	.178	**.163**	.436	.382	**.298.116.333.235**	.302	**.396.290**									
[17]	.208	**.480**	.077	.143	.131	.397	.411	.052	**.116**	.186	.111	.031	**.447.394**	.169	.112	.226	.170	**.369**	.300	.227	

to produce a left-right symmetric model. These design choices end up with 400 to 1000 components, depending on the category.

Table 2 illustrates the power of individual features and their combinations. The mean average precisions(mAP) of all categories on VOC 2010 val are shown. Note that features play different roles in different object categories. Furthermore, feature combination significantly improves performance on all categories.

Table 3 compares the results of our full system with leading approaches on the VOC 2010 test set. Our results are highly competitive to the leading performance on this benchmark.

6.2 Multi-Component vs. Monolithic vs. Per Exemplar Models

In addition to knowing where our approach stands in the detection field, we are also interested in knowing how much we benefit from our multi-component scheme. Table 4 illustrates control experiments that compare our multi-component model with a monolithic model using the same set of features (SIFT and poselet activations) as well as training data (the positive set of the monolithic model is the set of all positive data used in the component models). We conclude from the table that our multi-component model handles intra-class variations better than the monolithic model and therefore yields much improved results for all PASCAL categories.

On the other hand, our results are significantly better than [17] which uses a single object per component as positive set. This illustrates the generalization power of our component models through global object alignment.

Fig. 6. Visualization of our detection results on PASCAL VOC dataset. Red bounding boxes indicate detection. Figures on the left show correct detection, while figures on the right show some failure cases. Many are due to heavy occlusion/trancation, poor localization, and confusion between similar categories.

6.3 Keypoints and Mask Transfer via Component Models

Like [11,17], our multi-component models provide more information than just bounding boxes in the object detection framework. The output scores of the first-layer component classifiers imply the most similar component in appearance and configuration to a detected object. We refer to the one assigning the highest score to a detection as the "matching" component of that detection. See Figure 7 for some examples. Since training objects are tight within a component, a projection of the keypoints and mask of the seed object onto the test image based on the transformation between the seed and the detection windows provides reasonable estimates of the keypoint locations and the mask of the detected object. This is a straight-forward yet useful application which is difficult to obtain from most previous related work.

Fig. 7. Our multi-component models enable fine-grained visual recognition applications such as keypoint prediction and segmentation, as shown in the figures above. Each detected object in the test image (shown in the top-right of each category) is associated with a "matching" component that assigns the highest detection score to the object. The two figures on the left of each category depict the seed object (with its keypoints marked in blue) and the average mask of the "matching" component. Inference on keypoint locations and mask of the test object is obtained through a transformation between the bounding box of the seed and that of the test object, as well as bottom-up segmentation cues. The two figures on the right of each category show our results, where estimated keypoint locations are marked in pink, and segmentation mask in red.

In fact, further improvement on object mask prediction can be achieved through two modifications on the projection pipeline. First, the average mask of all aligned objects of the component is a more robust measurement of component mask than that of the seed. Second, the bottom-up image segmentation cues can be used to refine our results. Suppose that the final object mask is a selected union of the image segments, our objective is to maximize the amount of overlap of the final object mask to the transformed mask of the matching component. We use a greedy selection solution to approximate the result. Starting with the single best segment among the pool that overlaps with the transformed mask, we pick a segment from the pool at each iteration and merge it with the current mask so that it maximizes the intersection-over-union score. The iteration continues until the score starts to decrease. In practice, this approximation works very well, and Figure 7 highlights our mask prediction results.

7 Conclusion

This paper presents a novel multi-component model that, combined with object window selection, achieves highly competitive results for object detection. Each component characterizes a pose or subcategory of an object category, and objects within each component are tight in appearance and configuration. Therefore, component models are both easy to learn and highly discriminative. A second layer classifier is learned to aggregate the outputs of component models into final scores. We also illustrate applications of our multi-component models beyond detection, e.g., object keypoint and mask prediction.

Acknowledgments. We thank ONR MURI N00014-10-10933 for their support to this work. This research was conducted as part of Chunhui Gu"s Phd thesis at UC Berkeley.

References

1. Alexe, B., Deselaers, T., Ferrari, V.: What is an Object? In: Computer Vision and Pattern Recognition (2010)
2. Andrews, S., Tsochantaridis, I., Hofmann, T.: Support Vector Machines for Multiple-instance Learning. In: Neural Information Processing Systems (2002)
3. Arbeláez, P., Hariharan, B., Gu, C., Gupta, S., Bourdev, L., Malik, J.: Semantic Segmentation Using Regions and Parts. In: Computer Vision and Pattern Recognition (2012)
4. Bourdev, L., Malik, J.: Poselets: Body Part Detectors Trained Using 3D Human Pose Annotations. In: International Conference on Computer Vision (2009)
5. Bourdev, L., Maji, S., Brox, T., Malik, J.: Detecting People Using Mutually Consistent Poselet Activations. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part VI. LNCS, vol. 6316, pp. 168–181. Springer, Heidelberg (2010)
6. Brox, T., Bourdev, L., Maji, S., Malik, J.: Object Segmentation by Alignment of Poselet Activations to Image Contours. In: Computer Vision and Pattern Recognition (2011)

7. Dalal, N., Triggs, B.: Histograms of Oriented Gradients for Human Detection. In: Computer Vision and Pattern Recognition (2005)
8. Endres, I., Hoiem, D.: Category Independent Object Proposals. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part V. LNCS, vol. 6315, pp. 575–588. Springer, Heidelberg (2010)
9. Everingham, M., Van Gool, L., Williams, C., Winn, J., Zisserman, A.: The Pascal Visual Object Classes (VOC) Challenge. International Journal of Computer Vision 88(2), 303–338 (2010)
10. Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan, D.: Object Detection with Discriminatively Trained Part Based Models. Transactions on Pattern Analysis and Machine Intelligence 32(9), 1627–1645 (2010)
11. Gu, C., Lim, J., Arbeláez, P., Malik, J.: Recognition Using Regions. In: Computer Vision and Pattern Recognition (2009)
12. Gu, C., Ren, X.: Discriminative Mixture-of-Templates for Viewpoint Classification. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part V. LNCS, vol. 6315, pp. 408–421. Springer, Heidelberg (2010)
13. Lazebnik, S., Schmid, C., Ponce, J.: Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In: Computer Vision and Pattern Recognition (2006)
14. Li, F., Carreira, J., Sminchisescu, C.: Object Recognition as Ranking Holistic Figure-Ground Hypotheses. In: Computer Vision and Pattern Recognition (2010)
15. Lin, Y., Cao, L., Lv, F., Zhu, S., Yang, M., Cour, T., Yu, K., Huang, T.: Large-scale Image Classification: Fast Feature Extraction and SVM Training. In: Computer Vision and Pattern Recognition (2011)
16. Maji, S., Berg, A., Malik, J.: Classification Using Intersection Kernel Support Vector Machines is Efficient. In: Computer Vision and Pattern Recognition (2008)
17. Malisiewicz, T., Gupta, A., Efros, A.: Ensemble of Exemplar-SVMs for Object Detection and Beyond. In: International Conference on Computer Vision (2011)
18. Parkhi, O., Vedaldi, A., Jawahar, C., Zisserman, A.: The Truth About Cats and Dogs. In: International Conference on Computer Vision (2011)
19. Russakovsky, O., Lin, Y., Yu, K., Fei-Fei, L.: Object-Centric Spatial Pooling for Image Classification. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part II. LNCS, vol. 7573, pp. 1–15. Springer, Heidelberg (2012)
20. van de Sande, K., Gevers, T., Snoek, C.: Evaluating Color Descriptors for Object and Scene Recognition. Transactions on Pattern Analysis and Machine Intelligence 32(9), 1582–1596 (2010)
21. van de Sande, K., Uijlings, J., Gevers, T., Smeulders, A.: Segmentation as Selective Search for Object Recognition. In: International Conference on Computer Vision (2011)
22. Vedaldi, A., Fulkerson, B.: VLFeat: An Open and Portable Library of Computer Vision Algorithms (2008), `http://www.vlfeat.org/`
23. Yu, K., Zhang, T., Gong, Y.: Nonlinear Learning Using Local Coordinate Coding. In: Neural Information Processing Systems (2009)
24. Zhu, L., Chen, Y., Yuille, A., Freeman, W.: Latent Hierarchical Structural Learning for Object Detection. In: Computer Vision and Pattern Recognition (2010)