Fine-Grained Recognition without Part Annotations

Jonathan Krause[1] Hailin Jin[2] Jianchao Yang[2] Li Fei-Fei[1]
[1]Stanford University [2]Adobe Research
{jkrause,feifeili}@cs.stanford.edu {hljin,jiayang}@adobe.com

Abstract

Scaling up fine-grained recognition to all domains of fine-grained objects is a challenge the computer vision community will need to face in order to realize its goal of recognizing all object categories. Current state-of-the-art techniques rely heavily upon the use of keypoint or part annotations, but scaling up to hundreds or thousands of domains renders this annotation cost-prohibitive for all but the most important categories. In this work we propose a method for fine-grained recognition that uses no part annotations. Our method is based on generating parts using co-segmentation and alignment, which we combine in a discriminative mixture. Experimental results show its efficacy, demonstrating state-of-the-art results even when compared to methods that use part annotations during training.

Figure 1. In fine-grained recognition, categories share similar shapes, which allows for alignment to be done purely based on segmentation.

1. Introduction

Models of fine-grained recognition have made great progress in recognizing an ever-increasing number of categories. Performance on one standard dataset [44] has increased from 10.3% [44] to 75.7% [6] in only three years. On the data side, there has also been progress in expanding the set of fine-grained domains we have data for, which now includes *e.g.* birds [44, 47, 4], aircraft [42, 34], cars [41, 27, 32], flowers [35, 1], leaves [30], and dogs [25, 33].

Compared to generic object recognition, fine-grained recognition benefits more from learning critical parts of the objects that can help align objects of the same class and discriminate between neighboring classes [3, 16, 52, 10, 13]. Current state-of-the-art results are, therefore, from models that require part annotations as part of the supervised training process [51, 6]. This poses a problem for scaling up fine-grained recognition to an increasing number of domains.

Towards the goal of training fine-grained classifiers without part annotations, we make an important observation. Fig 1 illustrates our idea. Objects in a fine-grained class share a high degree of shape similarity, allowing them to be aligned via segmentation alone. If we can align them early in the training process, we can learn the characteristic parts without the annotation effort.

In this work, we propose a method to generate parts which can be detected in novel images and learn which parts are useful for recognition. Our method for generating parts leverages recent progress in co-segmentation [22, 29] to segment the training images. We then densely align images which are similar in pose, performing alignment across all images as the composition of these more reliable local alignments. Despite using fewer annotations, our method is state-of-the-art on the competitive CUB-2011 dataset [44] when using a VGGNet [40] for feature extraction, is on par with current state-of-the-art even when using a weaker CaffeNet [23] architecture, and is furthermore able to generalize to fine-grained domains which do not have part annotations, establishing a new state-of-the-art on the cars-196 [27] dataset by a large margin.

The remainder of the paper is organized as follows: We review related work in Sec. 2 and describe our method for generating parts in Sec. 3. Our use of these parts for recognition is covered in Sec. 4. We present experiments and analysis on the CUB-2011 and cars-196 datasets in Sec. 5 and conclude with future work in Sec. 6.

2. Related Work

Fine-Grained Recognition. A variety of methods have been developed for differentiating between fine-grained categories. Though many early approaches [15, 49, 50, 48] did not use part annotations, their performance has been eclipsed by methods developed to explicitly take advantage of the structure present in fine-grained classes [3, 6, 51, 53, 13]. A few works have even gone beyond the use of 2D part annotations, aiming to get a full correspondence across images via a 3D representation [16, 27]. Still other methods explore fine-grained classification with a human in the loop at test time, *e.g.* the visipedia project [8, 43, 45], which is complementary to our approach.

Of the methods developed that do not use part annotations, there have been a few works philosophically similar to ours in the goal of finding localized parts or regions in an unsupervised fashion [15, 18, 10], with [18] and [10] more relevant. Gavves *et al.* [18, 19] segment images via GrabCut [37], and then roughly align objects by parameterizing them as an ellipse. Chai *et al.* [10] do a joint segmentation and DPM [17] model fitting, extracting features around each DPM part. In contrast to these works, our alignment model is computed densely and is the composition of easier alignments between similar images. We also perform the task of detection at test time and ultimately achieve better classification results than either.

Current state-of-the-art methods are Zhang *et al.* [51] and Branson *et al.* [6], which are both supervised at the level of parts during training. Both employ a part detection model, with [51] generalizing the R-CNN framework [20] to detect parts in addition to the whole object, and [6] training a strongly-supervised deformable part model in a structured learning framework [5]. Of these two, our method is more related to [51] in that we use an R-CNN model for detection, but unlike either our method is completely unsupervised at the level of parts.

Co-segmentation. Co-segmentation, the task of segmenting the object common to a set of images, has made great strides in recent years [22, 38, 24, 9, 11, 10, 29]. Co-segmentation has even seen some success in fine-grained recognition [9, 11, 10], exploiting the low intra-class variability of fine-grained classes. Unlike [9, 11, 10], our co-segmentation approach follows a graph-cut approach, inspired by Guillaumin, Kuettel, and Ferrari [22]. The main difference between our approach and [22] is that bounding boxes are available during training in our problem setting. We use this to significantly improve segmentation quality via a refinement step that finds a segmentation covering the bounding box well. This fixes many common failure modes typical of a graph-cut segmentation framework. Similar intuition with a more sophisticated approach can be found in [31]. In our setting no ground truth segmentations are available, unlike the full model of [22], so we are thus more related to their "image+class" model.

3. Generating Parts

At the core of our approach for generating parts is the concept of *alignment by segmentation*, the process of aligning images via aligning their figure-ground segmentation masks. The key insight is that, even for complicated and deformable objects such as birds, a figure-ground segmentation (Fig. 2(b)) is often sufficient in order to determine an object's pose and localize its parts, as demonstrated in Fig. 1. We decompose the process of aligning all images as aligning pairs of images with similar poses, which we represent in a graph (Fig. 2 (c)), producing a global alignment (Fig. 2(d)) from these easier, local alignments. Based on these alignments we sample points across all images, which each determine a part (Fig. 2(e)).

3.1. Co-segmentation

In order to do alignment by segmentation, one must first establish a figure-ground segmentation of each image, which we do via co-segmentation (Fig. 2(b)). Co-segmentation is particularly attractive for fine-grained recognition because the appearance variation within each class is relatively small. Traditional co-segmentation approaches [38, 24] typically assume that no information besides object presence is available for each image, while in the setting of fine-grained recognition we also have bounding boxes available in these training images. Our approach effectively and efficiently uses this information.

Formulation. Our optimization for co-segmentation is inspired by Guillaumin *et al.* [22] and uses a GrabCut [37]-like approach at its base. Let θ_f^i be a foreground color model for image i, represented as a Gaussian mixture model, θ_b^i a similar background model, and θ_f^c a shared foreground color model for class c. The binary assignment of pixel p in image i to either foreground or background is denoted x_p^i, its corresponding RGB value is z_p^i, the set of segmentation assignments across all images is X, and p_f is a pixelwise foreground prior which we describe shortly. Our co-segmentation objective is:

$$\max_{X,\theta} \sum_i \left(\sum_p E(x_p^i, \theta^i, \theta_f^c; p_f^i) + \sum_{p,q} E(x_p^i, x_q^i) \right) \quad (1)$$

where

$$\begin{aligned} E(x_p^i, \theta^i, \theta_f^c; p_f^i) &= (1 - x_p^i) \log(p(z_p^i; \theta_b^i)) \\ + \frac{x_p^i}{2} \left(\log(p(z_p^i; \theta_f^i)) + \log(p(z_p^i; \theta_f^c)) \right) &+ E(x_p^i; p_f), \end{aligned} \quad (2)$$

$$E(x_p^i; p_f) = \begin{cases} \log(p_f) & x_p^i = 1 \\ \log(1 - p_f) & x_p^i = 0 \end{cases}, \quad (3)$$

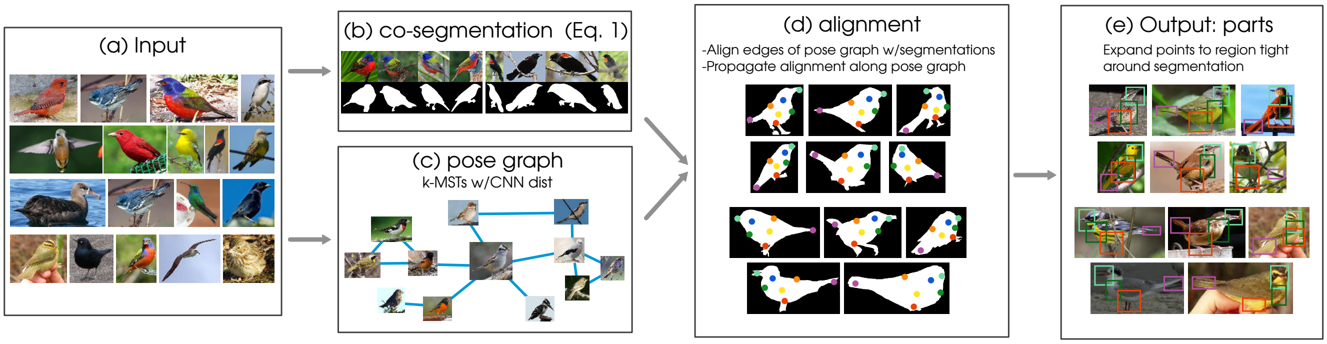

Figure 2. An overview of our method to generate parts used for recognition. We begin by segmenting all images in the training set with a co-segmentation approach (Sec. 3.1) and finding a graph used to determine which images to align (Sec. 3.2). With these, we sample points across all images, which form the basis for generating parts used in recognition (Sec. 4).

and $E(x_p^i, x_q^i)$ is the standard pairwise term between pixels p and q for a GrabCut [37] segmentation model, enforcing consistency between neighboring pixels with respect to their assigned binary foreground/background values. If $p_f = .5$ and $\theta_i^f = \theta_i^c$ then this is equivalent to GrabCut, and if only $p_f = .5$ then it reduces to the "image + class" model of [22] without the learned per-term weights.

Optimization is performed separately for each fine-grained class c, and proceeds by iteratively updating the appearance models θ_f^i, θ_b^i, θ_f^c and optimizing the foreground/background masks x^i. As is standard in a GrabCut formulation, we initialize the appearance models using the provided bounding boxes, with the pixels inside each bounding box marked foreground and the rest as background. This initial background remains fixed as background throughout the optimization.

Foreground Refinement. Though we have already used the bounding boxes available in fine-grained training sets in a standard GrabCut[37] fashion, we have not fully exploited their usefulness. As noted by [31], objects in bounding boxes typically occupy a non-negligible portion of the bounding box, and we can use this knowledge to significantly improve segmentation quality. Formally we represent this as constraints that a foreground segmentation must occupy between ω_1 and ω_2 of the total area of its bounding box and span at least ρ of its width and height. To satisfy these constraints, we perform a binary search over the pixelwise foreground prior p_f (Eq. 3) on a per image basis after the initial segmentation until the constraints are satisfied, initializing with $p_f = .5$. Since $p_f = 0$ produces an entirely background segmentation and $p_f = 1$ corresponds to an entirely foreground segmentation, this search will produce a segmentation satisfying the constraints.

In our experiments we set $\omega_1 = 10\%$, $\omega_2 = 90\%$, and $\rho = 50\%$, representing a weak set of constraints that is satisfied in 99.97% of the images in CUB-2011 [44]. As we demonstrate in our experiments and visualize in Fig. 3, this weak prior dramatically improves segmentation results, and

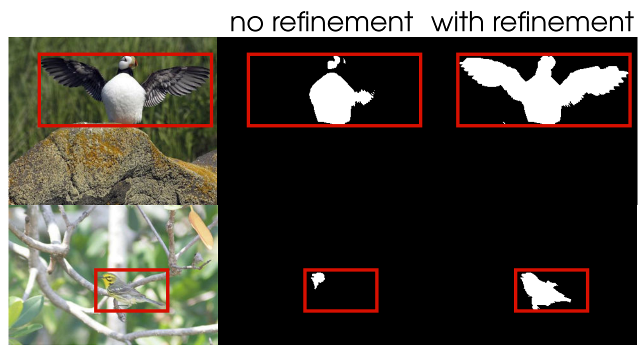

Figure 3. Refinement by searching for a foreground prior p_f satisfying weak bounding box-level constraints can correct very large errors in segmentation.

tends to fix a common failure mode of GrabCut segmentation methods of over- or under-segmenting images. More qualitative results are given in the supplementary material. We also note that most initial segmentations already satisfy these constraints, so this extra binary search does not significantly change the running time of co-segmentation, and is much cheaper than more sophisticated methods [31].

3.2. Choosing Images to Align

Aligning two objects with arbitrary poses remains a hard problem in computer vision. However, when given many instances of the same category, any single object is likely to have a similar pose with at least one other instance. This motivates our approach of decomposing the global task of aligning all training images into many smaller, simpler tasks of aligning images containing objects of similar poses. We formalize this requirement as building a connected graph $\mathcal{G}$ of images $\{I_i\}_{i=1}^n$ where each edge (I_i, I_j) is between two images containing objects of similar poses to be aligned. To reduce the variance in alignment, we furthermore require that each image $I_i \in \mathcal{G}$ be connected to at least k other images, aggregating all image to image alignments from the neighbors of I_i to increase robustness. Because $\mathcal{G}$ represents a graph of pose similarity, we refer to it as a *pose graph* (Fig. 2 (c)).

Figure 4. Nearest neighbors with conv_4 features, which tend to preserve pose.

How can we determine which images contain objects of similar poses without part annotations and without attempting the comparatively expensive process of aligning them in the first place? We do this with a simple, yet effective heuristic: as a proxy for difference in pose we measure the cosine distance between fourth-layer convolutional (conv_4) features around each bounding box, using a CNN pretrained on ILSVRC 2012 [23, 28, 39]. These intermediate features tend to be fairly robust to changes in *e.g.* background while maintaining information about pose; features in earlier layers are not as robust and features in later layers become too class-specific, eschewing pose information in favor of discriminative power. We have also experimented with using our segmentations from Sec. 3.1 and other feature representations to measure pose similarity, but have found very little difference in performance from simply using these CNN features. Fig. 4 shows examples of the nearest neighbors calculated with this metric, with more examples included in the supplementary material.

Pose Graph Construction. Using this distance metric we construct $\mathcal{G}$ satisfying the constraints by iteratively computing disjoint minimum spanning trees of the images, merging the trees into a single graph. Concretely, we decompose the pose graph as $\mathcal{G} = \bigcup_{i=1}^{k} M_i$, where M_1 is the minimum spanning tree of the dense graph $\mathcal{G}_D$ on all n images with edge weights given by cosine distance, and M_j is the minimum spanning tree of $\mathcal{G}_D \setminus \bigcup_{i=1}^{j-1} M_i$, which can be computed by setting the weights of all edges used in $M_1, \ldots, M_{j-1}$ to infinity. Since minimum spanning trees are connected, $\mathcal{G}$ is connected, and since $\mathcal{G}$ is composed of k disjoint minimum spanning trees, each node in $\mathcal{G}$ is connected to at least k other vertices, satisfying the constraints.

3.3. Aligning All Images

Given a pose graph connecting images with similar poses, what is the right way to use its structure to create an alignment between all images? In our approach, we first sample a large set of points in one image, representing the overall shape of an object, and then propagate these points to all images using the structure of the graph.

We start by sampling a set of points of size k_1 on the segmented foreground of a single image I_r, which we choose to be the image with the highest degree in $\mathcal{G}$. Then, while there is still at least one image that the points have not been propagated to, we propagate to the image I_j adjacent to the largest number of images in $\mathcal{G}$ which have already been propagated to. Let $\tau_{i,j} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ be a dense alignment function mapping a point in image i to its corresponding point in image j, which is learned based on the segmentations for I_i and I_j. Then, to propagate each of the k_1 points, we use $\tau_{i,j}$ to propagate the corresponding point from each image I_i adjacent to I_j in $\mathcal{G}$ and aggregate these separate propagated points via an aggregation function α, which we take to be the median of points propagated from each adjacent image. In this work we learn each dense alignment function $\tau_{i,j}$ with shape context [2], which we make robust to horizontal flips by additionally optimizing over the choice of flipping one of the images.

3.4. From Alignment to Parts

After globally aligning all images, the problem remains of using the alignment to generate parts for use in recognition. Without any part annotations, at this point it is not possible to tell which parts are semantically meaningful or useful for classification, so we instead target our part generation at producing a *diverse* set of parts. Specifically, we select a subset of the propagated points of size k_2 to be expanded into parts for recognition. We do this by clustering the trajectories of the k_1 points across all images, *i.e.* we represent each point i by its $2 \times n$-dimensional trajectory across all images, then cluster each of these trajectories via k-means into k_2 clusters, providing a good spread of points across the foreground of each image (Fig. 2 (d)).

We generate a single part from each one of these k_2 points by taking an area around each point with a fixed size with respect to the object's bounding box, then shrink the region until it is tight around the estimated segmentation (Fig. 2 (e)), yielding a tight bounding box around each generated part in each training image.

4. Using Parts for Recognition

Given a set of generated parts, what is the right way to use them for recognition, and how can they be found in novel images which do not even contain object-level bounding boxes? For classification, we propose an approach to discriminatively learn a mixture of parts, and for detection we compare two approaches which both can find our generated parts in novel images.

4.1. Discriminative Combination of Parts

An effective fine-grained classifier must reason about information at all parts, combining multiple sources of visual information. The most straightforward approach, shared by

Figure 5. Not all parts of an object are equally useful for recognition. Some, such as the legs, are only rarely useful, while others, like the head, contain most information useful for discrimination.

[51], [6], and most prior approaches, is to concatenate features at each part into one large vector and train a single classifier. However, doing so ignores the observation that not all parts are equally useful for recognition (Fig. 5). Motivated by this, we propose a more nuanced approach. Our approach is inspired by the max-margin template selection method of [12], originally used for visual font recognition.

Let f_p^i be features for image i at part p, and let $w_{p,c}$ be classification weights for part p and class c, learned for each part independently. Our goal is to learn a vector of (k_2+1)-dimensional weights v satisfying:

$$\min_v \sum_{i=1}^{n} \sum_{c \neq c^i} \max\left(0, 1 - v^T u^i_{c^i,c}\right)^2 + \lambda \|v\|_1 \tag{4}$$

where the p-th element of $u^i_{c^i,c}$ is the difference in decision values between correct class c^i and incorrect class c:

$$u^i_{c^i,c}(p) = (w_{p,c^i} - w_{p,c})^T f_p^i \tag{5}$$

This is equivalent to a one-class SVM (an SVM with only positive labels) with an L_2 loss and L_1 regularization, and can thus be solved efficiently by standard SVM solvers. Intuitively, this optimization tries to select a sparse weighing of classifiers such that, combined, the decision value for the correct class is always larger than the decision value for every other class by some margin, forming a discriminative combination of parts. Decision values for each u^i can be calculated via cross-validation while training the independent classifiers at each part. In comparison to [12], the main difference is that our formulation operates directly on decision values rather than the probabilities output by the template system in [12]. The final classification is given by:

$$\arg\max_c \sum_{p=1}^{k_2} v_p w^T_{p,c} f_p \tag{6}$$

4.2. Finding Parts at Test Time

The other main challenge in using our automatically generated parts is finding them in novel, completely unannotated images. We experiment with two different methods, one based on [51] and the other a direct extension of our part generation method, applied to bounding box-level object detections.

Part Detectors. Our first method for locating parts at test time involves training dedicated part detectors, and is based on the Δ_{box} method of Zhang *et al.* [51], originally intended for use with ground truth part locations: an R-CNN [20] is trained for the entire object and every part by treating each as a separate category. At test time, all detectors are run on an image, each detection score is transformed into a probability via a sigmoid function, and the joint configuration of bounding box and parts is scored as the product of probabilities, with part detections that do not fall within 10 pixels of the bounding box set to probability 0. Since our set of generated parts is potentially much larger in size than the set of parts considered in [51], we change the joint configuration scoring method to normalize the log-probabilities from the part detectors by $\frac{1}{k_2}$. This gives the information at local parts and the global object equal weight, robustly combining the two. Our other difference from [51] is training each R-CNN with bounding box regression, which improves detection AP on CUB-2011 from 88.1 to 92.9.

Test-Time Segmentation and Alignment. Our second method is a direct extension of the segmentation and alignment done on the training images. We first do detection with an R-CNN trained on the whole bounding box, then use this predicted bounding box in our segmentation framework of Sec. 3.1, removing the foreground class appearance term since the class label is unknown at test time. The nearest neighbors of the test image in the training set are calculated using conv_4 features and alignment from those images is done exactly as described in Sec. 3.2. This method has an advantage over the part detector method in that an R-CNN only needs to be trained for the whole object, rather than $k_2 + 1$ categories, and it also produces a segmentation of test images, which may be useful for other applications, but it has the disadvantage of being somewhat slower at test time due to the alignment and segmentation steps. In all experiments with this method, we use 5 nearest neighbors.

5. Experiments

5.1. Datasets

We evaluate on the CUB-2011 [44] and cars-196 [27] datasets. CUB-2011 contains 11,788 images of 200 species of birds, and is generally considered the most competitive dataset within fine-grained recognition, while cars-196 has

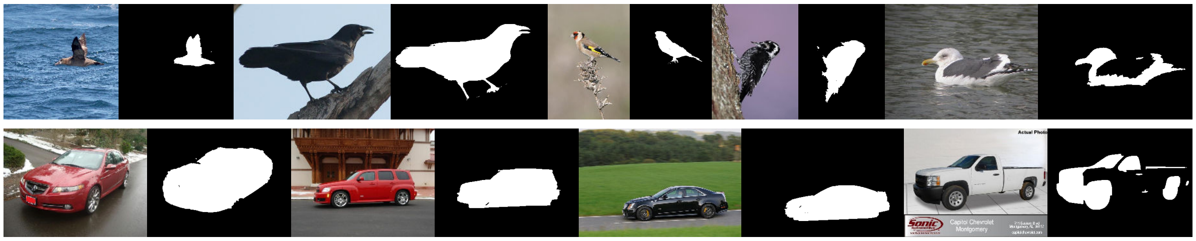

Figure 6. Example segmentations from our co-segmentation method on CUB-2011(top) and cars-196(bottom). The last image in each row is a failure case.

Method	Jac. Sim.
GrabCut [37]	70.84
+class ≈ [22]	67.78
+refine	74.72
+class+refine	75.47

Table 1. Co-segmentation results on CUB-2011 as measured by Jaccard similarity of the ground truth foreground with the predicted segmentation. GrabCut is equivalent to our model without the class foreground term or foreground prior refinement, which we add in "+class" and "+refine", respectively, with "+class+refine" corresponding to our full co-segmentation model.

16,185 images of 196 car types. Both datasets have a single bounding box annotation in each image, and CUB-2011 moreover contains rough segmentations and 15 keypoints annotated per image, which we do not use in our algorithm.

5.2. Implementation Details

For all R-CNN training we use fc_6 features from a network trained on ILSVRC2012 [39] with no fine-tuning and 16 pixels of padding around each bounding box or part. The R-CNN takes the majority of running time, at about 20 sec./image. Methods without fine-tuning also use fc_6 features, extracted on each part with 16 pixels of padding. All features are extracted using Caffe [23]. R-CNNs are trained with the default network (pre-trained on ILSVRC2012 and fine-tuned on PASCAL). Features for pose graph construction are from a pre-trained CaffeNet [23], which is similar to [28]. When fine-tuning networks in our experiments, two separate networks are fine-tuned: one for the whole object and one for the set of all generated parts. All regularization parameters and the weights for the discriminative combination of parts for fine-tuned networks are determined based on non-fine-tuned features, since fine-tuning makes training decision values overconfident. For part generation we set $k = 5$, $k_1 = 500$, $k_2 = 31$, and use a maximum part size of 25% of the geometric mean of the bounding box dimensions around each keypoint. We defer other implementation details to our code, available on the first author's website.

5.3. Co-segmentation Results

We first perform an analysis of our co-segmentation method, evaluated on CUB-2011. Results are given in Tab. 1. Interestingly, adding in the class foreground appearance term hurts a pure GrabCut approach, but helps when a refinement step is added in. This happens because with only the additional class term, the foreground model underfits, tending to result in an undersegmentation. However, when the refinement step is added, the learned class term provides for a strong refinement initialization, allowing the per-image term to fit the foreground more accurately. This result highlights the different approach needed for co-segmentation when given bounding boxes, as class appearance terms help substantially in the no bounding box case [22]. Tab. 1 also demonstrates the importance of the foreground prior refinement, improving upon GrabCut by nearly 4%. Fig. 6 shows qualitative results on CUB-2011 and cars-196, with more results given in the supplementary material. Our approach is generally able to segment the foreground object well, but understandably has trouble when the foreground and background are too similar.

5.4. Recognition Results

We first perform a detailed analysis of our method on CUB-2011 before moving on to compare to other methods on both CUB-2011 and cars-196.

Method Analysis. Tab. 2 details many variants of our method, using the fc_6 layer of a CaffeNet [23] for feature extraction. We observe that the part detector method works better than the test-time segmentation method, performing better under both part combination strategies. This indicates that the learned part detectors are able to generalize well to unseen images, and motivates our use of the part detector method for the rest of the analysis. Second, combining the parts discriminatively is always better than the concatenation strategy. We note that PD+DCoP, without bounding boxes during test time and without fine-tuning, is already able to out-perform a fine-tuned CNN when given the ground truth bounding box, highlighting the importance of reasoning at the level of parts and validating the effectiveness of our approach.

We visualize the parts with the highest weights in the discriminative combination of parts in Fig. 7(left) and observe that they both fire consistently and represent a diverse set of parts, with the top two parts (bird heads with varying amounts of context, extremely discriminative on their own) excepted. We furthermore show example images in

Figure 7. The top parts chosen by our method, excepting the whole object "part", visualized by the highest scoring detections in the test set. Each row is a different part. Shown at left are our top parts for CUB-2011 and at right are the top parts for cars-196.

Method	Acc.
R-CNN [20]	58.8
R-CNN+ft	65.3
CNN+GT BBox	61.3
CNN+GT BBox+ft	67.9
TS+concat	63.4
TS+DCoP	68.5
PD+concat	67.6
PD+DCoP	69.7
PD+DCoP+flip	71.1
PD+DCoP+flip+ft	73.7
PD+DCoP+flip+GT BBox	72.4
PD+DCoP+flip+GT BBox+ft	74.9

Table 2. Analysis of different variants of our method on CUB-2011. R-CNN refers to training an R-CNN [20] for birds generically and extracting features on the whole bounding box. "+ft" means that the CNN used to extract features after detection was fine-tuned for classification. "PD" refers to using the generated parts in a part detection framework, and "TS" refers to the method of doing segmentation at test time and aligning the test image with a set of training images (Sec. 4.2). "concat" and "DCoP" are the two methods of combining multiple parts, and refer to concatenating the features and the discriminative combination of parts (Sec. 4.1), respectively. "+flip" indicates training and testing with both original and horizontally flipped images, averaging the decision values during test, and "+GT BBox" indicates giving the method oracle bounding box information. Performance is measured with 200-way accuracy.

which our method classifies correctly but an R-CNN with fine-tuning is incorrect in Fig. 8. Even in cases of unusual pose or inaccurate detections, our method is still able to accurately localize one or more parts and classify correctly, while the R-CNN is forced to reason at a whole bounding box level, unable to discriminate the fine-grained classes.

Adding in flipped images improves performance by another 1.4%, and fine-tuning improves that further by 2.6%. Granting our method the ground truth bounding box at test time improves results by only 1.3% (1.2% with fine-tuning), suggesting that improvements to the detection model will not impact classification results much on CUB-2011.

Method	CNN Used [23]	CNN Used [40]
R-CNN [20]	58.8	69.0
R-CNN+ft	65.3	72.5
CNN+GT BBox	61.3	70.0
CNN+GT BBox+ft	67.9	75.0
PD+DCoP+flip	71.1	78.8
PD+DCoP+flip+ft	73.7	82.0
PD+DCoP+flip+GT BBox+ft	74.9	82.8

Table 3. Impact of CNN choice on variants of our method, measured in 200-way accuracy.

Impact of CNN Architecture. We also analyze the effect of network architecture used to extract features at each part, comparing CaffeNet [23] and VGGNet (the 19-layer configuration "E" from [40]) on a subset of method variants. Tab. 3 details the results. In all cases, using a VGGNet significantly improves results, so we present the remainder of the results using a VGGNet for feature extraction.

Comparison to Prior Work on CUB-2011. In Tab. 4 we compare our method to prior work on CUB-2011, listing the amount of annotation each method uses. Our method is best by a large margin among methods which use no annotations at test time, and even outperforms all methods that use part annotations during training, only beaten by the variant of PN-DCN [6] which uses part annotations at test time (and tied with the variant of PB R-CNN that does the same). Using the weaker CaffeNet architecture, our results are within two percent of the variants of PB R-CNN [51] and PN-DCN [6] that use no annotations at test time but use part supervision during training. We anticipate that improving PB R-CNN and PN-DCN with better CNN architectures will again result in performance higher than our best approach due to their additional supervision. Expert human performance on CUB-2011 is roughly 93% [7].

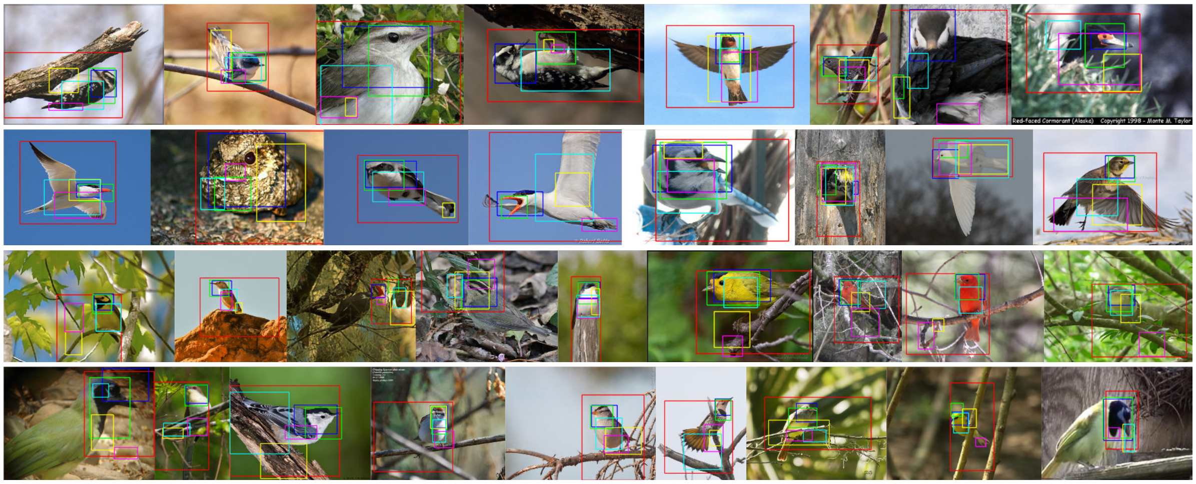

Figure 8. Test images in which our method is correct but an R-CNN with fine-tuning is incorrect, visualized with detections for the whole object and five parts with the highest weight in the discriminative combination of parts. The top two rows depicts images where birds have unusual poses, and the bottom two rows show cases where the detection is inaccurate.

Method	Train Anno.	Test Anno.	Acc.
Alignments [19]	*n/a*	*n/a*	53.6
Ours	**BBox**	***n/a***	**82.0**
GMTL [36]	BBox	BBox	44.2
Symbiotic [10]	BBox	BBox	59.4
Alignments [19]	BBox	BBox	67.0
Ours+BBox	**BBox**	**BBox**	**82.8**
PB R-CNN [51]	BBox+Parts	*n/a*	73.9
PN-DCN [6]	BBox+Parts	*n/a*	75.7
DPD [53]	BBox+Parts	BBox	51.0
POOF [3]	BBox+Parts	BBox	56.8
Nonparametric [21]	BBox+Parts	BBox	57.8
Symbiotic [10]	BBox+Parts	BBox	61.0
DPD+DeCAF [14]	BBox+Parts	BBox	65.0
PB R-CNN [51]	BBox+Parts	BBox	76.4
Symbiotic [10]	BBox+Parts	BBox+Parts	69.5
POOF [3]	BBox+Parts	BBox+Parts	73.3
PB R-CNN [51]	BBox+Parts	BBox+Parts	82.0
PN-DCN [6]	BBox+Parts	BBox+Parts	85.4

Table 4. Comparison of different methods on CUB-2011, sorted by amount of annotation used. "Ours" indicates our full model (PD+DCoP+flip+ft), and "Ours+BBox" grants our method the ground truth bounding box at test time. "Parts" refers to using any annotation at the level of parts at all. Since the exact amount of annotation used varies from work to work, we defer to the original sources for details.

Comparison on cars-196 The main advantage of our method is that it allows us to do accurate recognition on classes that do not have part annotations, scaling up to a larger number of fine-grained domains, which methods such as [6, 51, 3] cannot do. To this end, we compare performance on the cars-196 [27] dataset, with results given in Tab. 5. All results not reported by prior work use the VGGNet [40] architecture, with an architecture comparison in the supplementary material. Our method is able to greatly outperform all previously-reported results [13, 27, 46, 26], setting a new state-of-the-art by 18.7%. Parts chosen in the discriminative combination of parts are shown in Fig. 7. These parts tend to either contain information relating to either the general body type of cars (top two parts), or focus on fine details such as the grille or headlights.

Method	Train	Test	Acc.
R-CNN [20]	BBox	*n/a*	57.4
R-CNN+ft	BBox	*n/a*	88.4
Ours	**BBox**	***n/a***	**92.6**
CNN+GT BBox	BBox	BBox	59.9
BB [13]	BBox	BBox	63.6
BB-3D-G [27]	BBox	BBox	67.6
LLC [46]	BBox	BBox	69.5
ELLF [26]	BBox	BBox	73.9
CNN+GT BBox+ft	BBox	BBox	89.0
Ours+GT BBox	**BBox**	**BBox**	**92.8**

Table 5. Comparison of methods on cars-196 [27]. Performance is measured with 196-way accuracy.

6. Conclusion

In this work we have presented a method for fine-grained classification which does not require part annotations at training time, setting a new state-of-the-art on the CUB-2011 and cars-196 datasets. We believe that developing such methods will be important for scaling up fine-grained recognition to an ever-increasing number of visual domains, in which it will be cost-prohibitive to annotate parts.

Acknowledgements. We thank Olga Russakovsky, Serena Yeung, Andre Esteva, Jon Brandt, Scott Cohen, and Brian Price for helpful feedback. This work is partially supported by Adobe and an ONR-MURI grant.

References

[1] A. Angelova, S. Zhu, and Y. Lin. Image segmentation for large-scale subcategory flower recognition. In *Workshop on Applications of Computer Vision (WACV)*, pages 39–45. IEEE, 2013.

[2] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002.

[3] T. Berg and P. N. Belhumeur. Poof: Part-based one-vs.-one features for fine-grained categorization, face verification, and attribute estimation. In *Computer Vision and Pattern Recognition (CVPR)*, pages 955–962. IEEE, 2013.

[4] T. Berg, J. Liu, S. W. Lee, M. L. Alexander, D. W. Jacobs, and P. N. Belhumeur. Birdsnap: Large-scale fine-grained visual categorization of birds. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE.

[5] S. Branson, O. Beijbom, and S. Belongie. Efficient large-scale structured learning. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1806–1813, 2013.

[6] S. Branson, G. Van Horn, P. Perona, and S. Belongie. Improved bird species recognition using pose normalized deep convolutional nets. In *British Machine Vision Conference*, 2014.

[7] S. Branson, G. Van Horn, C. Wah, P. Perona, and S. Belongie. The ignorant led by the blind: A hybrid human–machine vision system for fine-grained categorization. *International Journal of Computer Vision*, pages 1–27, 2014.

[8] S. Branson, C. Wah, F. Schroff, B. Babenko, P. Welinder, P. Perona, and S. Belongie. Visual recognition with humans in the loop. In *European Conference on Computer Vision (ECCV)*, pages 438–451. Springer, 2010.

[9] Y. Chai, V. Lempitsky, and A. Zisserman. Bicos: A bi-level co-segmentation method for image classification. In *International Conference on Computer Vision (ICCV)*. IEEE.

[10] Y. Chai, V. Lempitsky, and A. Zisserman. Symbiotic segmentation and part localization for fine-grained categorization. In *International Conference on Computer Vision (ICCV)*, pages 321–328. IEEE, 2013.

[11] Y. Chai, E. Rahtu, V. Lempitsky, L. Van Gool, and A. Zisserman. Tricos: A tri-level class-discriminative co-segmentation method for image classification. In *European Conference on Computer Vision (ECCV)*, pages 794–807. Springer, 2012.

[12] G. Chen, J. Yang, H. Jin, J. Brandt, E. Shechtman, A. Agarwala, and T. X. Han. Large-scale visual font recognition. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2014.

[13] J. Deng, J. Krause, and L. Fei-Fei. Fine-grained crowdsourcing for fine-grained recognition. In *Computer Vision and Pattern Recognition (CVPR)*, pages 580–587, 2013.

[14] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. *arXiv preprint arXiv:1310.1531*, 2013.

[15] K. Duan, D. Parikh, D. Crandall, and K. Grauman. Discovering localized attributes for fine-grained recognition. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3474–3481, 2012.

[16] R. Farrell, O. Oza, N. Zhang, V. I. Morariu, T. Darrell, and L. S. Davis. Birdlets: Subordinate categorization using volumetric primitives and pose-normalized appearance. In *International Conference on Computer Vision (ICCV)*, pages 161–168. IEEE, 2011.

[17] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[18] E. Gavves, B. Fernando, C. G. Snoek, A. W. Smeulders, and T. Tuytelaars. Fine-grained categorization by alignments. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1713–1720, 2013.

[19] E. Gavves, B. Fernando, C. G. Snoek, A. W. Smeulders, and T. Tuytelaars. Local alignments for fine-grained categorization. *International Journal of Computer Vision*, pages 1–22, 2014.

[20] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition*, 2014.

[21] C. Goering, E. Rodner, A. Freytag, and J. Denzler. Nonparametric part transfer for fine-grained recognition. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2489–2496. IEEE, 2014.

[22] M. Guillaumin, D. Küttel, and V. Ferrari. Imagenet auto-annotation with segmentation propagation. *International Journal of Computer Vision (IJCV)*, pages 1–21, 2014.

[23] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[24] A. Joulin, F. Bach, and J. Ponce. Discriminative clustering for image co-segmentation. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1943–1950. IEEE, 2010.

[25] A. Khosla, N. Jayadevaprakash, B. Yao, and F.-f. Li. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, CVPR*. Citeseer, 2011.

[26] J. Krause, T. Gebru, J. Deng, L.-J. Li, and L. Fei-Fei. Learning features and parts for fine-grained recognition. In *International Conference on Pattern Recognition*, Stockholm, Sweden, August 2014.

[27] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*. IEEE, 2013.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems (NIPS)*, pages 1097–1105, 2012.

[29] D. Kuettel, M. Guillaumin, and V. Ferrari. Segmentation propagation in imagenet. In *European Conference on Computer Vision (ECCV)*, pages 459–473. Springer, 2012.

[30] N. Kumar, P. N. Belhumeur, A. Biswas, D. W. Jacobs, W. J. Kress, I. C. Lopez, and J. V. Soares. Leafsnap: A computer vision system for automatic plant species identification. In *European Conference on Computer Vision (ECCV)*, pages 502–516. Springer, 2012.

[31] V. Lempitsky, P. Kohli, C. Rother, and T. Sharp. Image segmentation with a bounding box prior. In *International Conference on Computer Vision (ICCV)*, pages 277–284. IEEE, 2009.

[32] Y.-L. Lin, V. I. Morariu, W. Hsu, and L. S. Davis. Jointly optimizing 3d model fitting and fine-grained classification. In *European Conference on Computer Vision (ECCV)*, pages 466–480. Springer, 2014.

[33] J. Liu, A. Kanazawa, D. Jacobs, and P. Belhumeur. Dog breed classification using part localization. In *European Conference on Computer Vision (ECCV)*, pages 172–185. Springer, 2012.

[34] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. Technical report, 2013.

[35] M.-E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 1447–1454. IEEE, 2006.

[36] J. Pu, Y.-G. Jiang, J. Wang, and X. Xue. Which looks like which: Exploring inter-class relationships in fine-grained visual categorization. In *European Conference on Computer Vision (ECCV)*, pages 425–440. Springer, 2014.

[37] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 309–314. ACM, 2004.

[38] M. Rubinstein, A. Joulin, J. Kopf, and C. Liu. Unsupervised joint object discovery and segmentation in internet images. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1939–1946. IEEE, 2013.

[39] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *arXiv preprint arXiv:1409.0575*, 2014.

[40] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[41] M. Stark, J. Krause, B. Pepik, D. Meger, J. J. Little, B. Schiele, and D. Koller. Fine-grained categorization for 3d scene understanding. In *British Machine Vision Conference (BMVC)*, September 2012.

[42] A. Vedaldi, S. Mahendran, S. Tsogkas, S. Maji, B. Girshick, J. Kannala, E. Rahtu, I. Kokkinos, M. B. Blaschko, D. Weiss, B. Taskar, K. Simonyan, N. Saphra, and S. Mohamed. Understanding objects in detail with fine-grained attributes. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[43] C. Wah, S. Branson, P. Perona, and S. Belongie. Multiclass recognition and part localization with humans in the loop. In *International Conference on Computer Vision (ICCV)*, pages 2524–2531. IEEE, 2011.

[44] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[45] C. Wah, G. Horn, S. Branson, S. Maji, P. Perona, and S. Belongie. Similarity comparisons for interactive fine-grained categorization. In *Computer Vision and Pattern Recognition (CVPR)*, 2014.

[46] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3360–3367. IEEE, 2010.

[47] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-ucsd birds 200. 2010.

[48] S. Yang, L. Bo, J. Wang, and L. G. Shapiro. Unsupervised template learning for fine-grained object recognition. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3122–3130, 2012.

[49] B. Yao, G. Bradski, and L. Fei-Fei. A codebook-free and annotation-free approach for fine-grained image categorization. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3466–3473. IEEE, 2012.

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

[51] N. Zhang, J. Donahue, R. Girshick, and T. Darrell. Part-based r-cnns for fine-grained category detection. In *European Conference on Computer Vision (ECCV)*, pages 834–849. Springer, 2014.

[52] N. Zhang, R. Farrell, and T. Darrell. Pose pooling kernels for sub-category recognition. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3665–3672. IEEE, 2012.

[53] N. Zhang, R. Farrell, F. Iandola, and T. Darrell. Deformable part descriptors for fine-grained recognition and attribute prediction. In *International Conference on Computer Vision (ICCV)*, pages 729–736. IEEE, 2013.