The Benefits and Challenges of Collecting Richer Object Annotations

Ian Endres, Ali Farhadi, Derek Hoiem, and David A. Forsyth
Department of Computer Science
University of Illinois Urbana Champaign
{iendres2,afarhad2,dhoiem,daf}@uiuc.edu

Abstract

Several recent works have explored the benefits of providing more detailed annotations for object recognition. These annotations provide information beyond object names, and allow a detector to reason and describe individual instances in plain English. However, by demanding more specific details from annotators, new difficulties arise, such as stronger language dependencies and limited annotator attention. In this work, we present the challenges of constructing such a detailed dataset, and discuss why the benefits of using this data outweigh the difficulties of collecting it.

1. Introduction

Object recognition is a fundamental problem of computer vision, and it is notorious for needing large amounts of annotation. While some works ask how to learn with less supervision [14, 8], a recent increase in availability of internet based annotators also lets us ask, what can we do with more data? Certainly we can collect more examples to train more reliable models for categorization, but we can also collect more detailed annotations for examples. These details could include the layout of an object, and descriptions of its underlying properties, such as pose, composition, or functionality.

Several works [12, 17, 9, 4] have already taken advantage of large annotation sources, and LabelMe [12] and LotusHill [17] are examples from two ends of the spectrum. LabelMe allows anyone to provide rough polygons that localize objects or scene elements, with no monetary reward. LotusHill provides rich object descriptions that range from object labels to constituent parts, segmentations, and other details. However, these finely detailed annotation requires expensive trained workers.

Mechanical Turk [1] provides a balance between the low cost of LabelMe and the high quality of LotusHill. Researchers can use Mechanical Turk to pay semi-qualified workers to produce detailed but not pixel-perfect annotations. When collecting increasingly more detailed annotations there are many inherent difficulties that arise from potential ambiguities and more demanding requirements of the tasks.

In this work, we first discuss the benefits of more detailed annotations, then we give an overview of the challenges of collecting this data and our solutions for many of the problems. Finally, we give a summary of the resulting datasets.

2. Benefits of Richer Annotation

Object recognition, typically posed as the problem of categorization, has made great progress in recent years. For many applications, categorization has only been a necessary stepping stone, where ultimately these applications need richer descriptions of the objects they see. If a robot detects an alligator, it should also know that it is dangerous, and identify which end can bite. Several threads of work have made strides to introduce richer representations along these lines. Human recognition has enjoyed most of the focus here, with pose [11] and face [9] recognition. However, recent work has also considered a richer description for objects in general as well.

Making semantic knowledge explicit in object representations allows recognition systems to describe, in words, what they see. Furthermore, such a representation can encode the necessary information to directly infer other properties that are not visually obvious, such as functionality. Lampert et al. [10] take the most basic approach and assigns semantic properties to each category, and it is assumed that each instance within a category has the same properties. The power of this representation is that these properties may be shared across categories, allowing predictions about unfamiliar objects. The annotation required here is comparable to annotating category labels, with the only additional

overhead of writing a table of properties for each category.

Works such as [9, 7] instead want to make distinctions between instances of objects, for faces or general domains of objects. This instance based learning requires more annotation, but it comes with a significant benefit. These methods can now give finer descriptions of each object they are presented. This allows them to direct attention to interesting or unusual properties of an object, such as "this boat is missing its sail". They can even produce plain English comparisons between two objects, such as "this face is more feminine than this other face." Each training example is labeled with a set of binary attributes, which represent some *semantics*, or properties that have some underlying meaning that can be expressed with words.

Curiously, by collecting more annotations for some objects and learning classifiers that generalize across categories, we need less data in the long run. Since this produces representations that are designed to be stable across categories, we can describe unfamiliar objects without any training examples, and even learn new category models with little or *no* training examples.

However, these binary annotations gives no spatial information, such as where an animal's legs are or the extent of its fur. Learning from these binary attributes is akin to using weakly annotated images for categorization, where the existence of the object is known, but not its location. While some works have addressed this problem for objects [8], having additional localized data can ease the learning problem. In particular, without spatial information, it is difficult to avoid relying on correlated or loosely related features. It is especially important for classifiers to learn to only use features directly related to a property when learning to generalize across categories. We begin exploring the use of spatial annotations in [6].

Including spatial information not only eases the learning problem, but it also opens opportunities to learn even richer object representations. This includes predicting explicit part configurations, which has already been explored for human recognition. In fact, Bourdev and Malik [3] recently showed that these part annotations can even improve detection performance for humans. From these part layouts, we can predict descriptions of an object's pose or the action it's performing. Furthermore, it may help improve segmentations to determine the extent and shape of an object. All of these abilities provide a better understanding of the objects found in images.

To facilitate future research of these semantic representations, we introduce the CORE (Cross-category Object REcognition) dataset [6], which intends to supersede the dataset of Farhadi et al. [7], which we refer to as ATT09.

3. Data Overview

Our dataset collection experience covers four types of data: images, binary attributes, polygon labels, and segmentation masks. We introduce each of these types here, and briefly motivate their uses.

Images: To collect images for the CORE dataset, we use pre-filtered images from ImageNet [4]. ImageNet provides images that are arranged by a hierarchy of object categories, and ensures with high probability that an image contains the category of interest. While ImageNet eases the difficulty of sifting through many irrelevant images, it is still necessary to remove many unsuitable images to obtain a representative and diverse set of images of objects found in the "wild."

Binary Attributes: The most basic annotation type is the binary attribute, which indicates the existence or lack of a certain property. These properties include anything that a person might mention when describing an object: its shape, constituent parts, compositional material, viewpoint, pose, or surrounding context. They are cheap to collect, and are helpful for describing properties without spatial information, such as an object's functionality or its overall state. The interface for labeling these can be seen in Figure 1a.

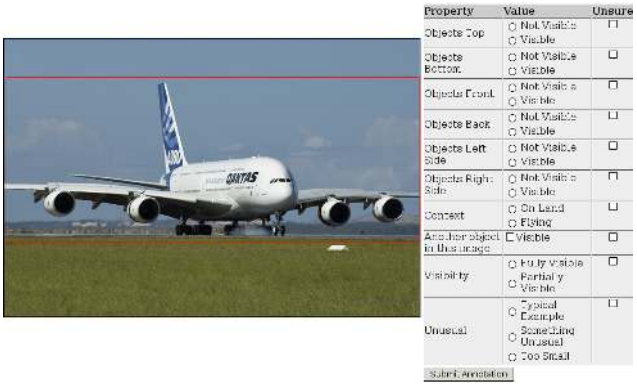
Polygon Labels: As discussed in section 2, there are many potential benefits from localized areas of interest. The interface can be seen in figure 1b. This flash based web interface was graciously provided by Alex Sorokin [15], and is freely available.

First, we obtain a polygon for each object, giving a rough outline that is more detailed than the bounding boxes provided in most datasets. Then, for each object, we have an annotator draw polygons for each part. These are the key component for learning a model that predicts an object's configuration. A fixed list of parts is provided to the annotator, and the list is determined by the category of the object to be labeled.

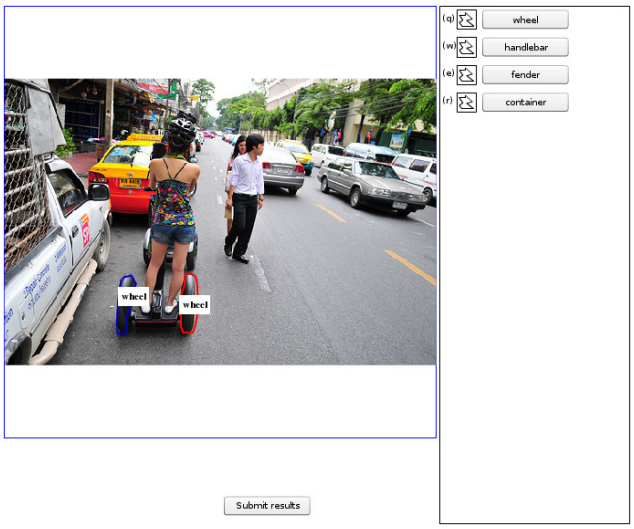
Material Segmentation Masks: The final type of annotation is a segmentation mask for materials. Being able to localize different materials restricts the possible locations and types of objects in an image. Furthermore, predicting the material of an unfamiliar object gives invaluable information about its place in a taxonomy of objects. We use a segmentation mask rather than a polygon, because materials can have holes, be disconnected, and have other unusual spatial behavior. An example of the annotation task can be seen in Figure 1c, again with the use of the annotation tool from [15].

4. Quality Issues

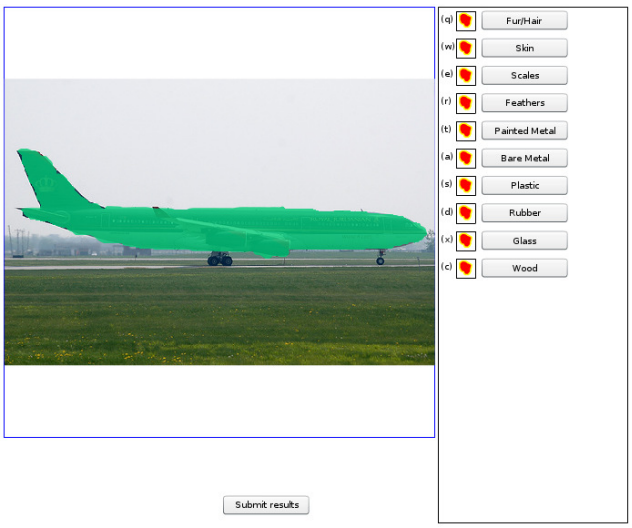
When collecting these detailed annotations, we found a characteristic set of difficulties, which stem from two main



(a) Binary Attributes



(b) Polygons



(c) Segmentation Masks

Figure 1. Example interfaces for each of the tasks. The binary interface (1a) includes an "unsure" option to flag potential confusion. The part list (1b) are tailored to the object of interest. In contrast, the list of materials includes any possible material found in images.

sources: the quality of the annotator and the complexity of the task. Each of these themes will recur throughout this section:

Annotators: The primary concerns with the annotators are their motivation and ability to understand the task. Furthermore, the annotators are not computer vision experts, and they are not aware of which mistakes are more costly to our machine learning techniques. Therefore even if they are well motivated they may not make the best decision when a subtle judgment is required. An example would be how to draw a polygon around an object that is partially occluded by a pole. While this can be addressed in the instructions, it is not possible to enumerate every special case, especially because annotators often ignore long detailed instructions (see Section 6.2).

Task Complexity: In addition to annotators, the task itself may cause confusion and present some difficulty. When asking for detailed annotations, the decision boundaries become less clear, such as how far should an object's torso extend, or are bats' wings covered in skin or fur. These can lead to inconsistent results. Furthermore, when asking for many annotations for a given image, the annotators may begin to miss details, simply because they cannot attend to every detail.

4.1. Collecting Images

The first hurdle of any dataset is collecting a diverse set of images that not only facilitate learning robust models, but also thoroughly evaluate the generalization capabilities of the learned model. For example, relying on a single source for images of cars, such as Flickr [2], can introduce a skewed view of cars, the majority of which come from car shows or race tracks, where the cars are exotic, interesting, and rarely found in a natural street scene. In general, a dataset of images has to avoid the following pitfalls:

Canonical Poses: Humans are often photographed from the front, and cars from the side and front. Any robust model should be insensitive to changes in viewpoint, and the dataset should encourage this with a variety of poses and viewpoints.

Archetypal Examples: As mentioned with the Flickr car bias, we want to avoid collections that do not fully represent the objects as they are found "in the wild".

Foreground Only: It is typically helpful to avoid images that contain only an object, without a surrounding scene, which is a common criticism of older datasets. Evaluating with many of these images won't fully evaluate the ability to localize, and a system could learn an artificial spatial prior. Furthermore, learning from scene based images may allow taking advantage of contextual information.

Stylized Images: Many images with solid backgrounds and edited colormaps or appearance can also cause difficulties. These types of images are more likely for inanimate objects such as household items, which introduces a clear bias.

Annotators of ImageNet were not made aware of these criteria, since their goal was to simply filter the images, so we carefully select images by hand to avoid the above issues.

4.2. Binary Attributes

Given an object, the annotator chooses whether a set of binary properties holds for each object. To handle cases where a particular attribute does not make sense, or there is some unexpected confusion, we add an "unknown/unsure" checkbox. This gives the annotator an opportunity to assert their uncertainty, allowing us to identify difficult attributes without having to reject conflicting annotations. For training, these "unsure" labels can be used to ignore potentially unhelpful training examples.

The main source of labeling inconsistency is the viewpoint attribute. This attribute indicates which portion of the object is visible. The main difficulty here appears to be communicating the task to the annotator in simple English. However, results improved significantly between after updating the definition of the task. The first attempt asked the annotator to indicate which way the object was facing, defined by the "canonical forward facing direction" with respect to the camera. For the first airplane in Figure1a, the label would be "into the camera, and to the right". Clearly this description can be quite confusing. We get much better results by updating the instructions to ask for which sides of the object are visible, such as the front, and right side, in Figure 1a.

Another interesting source of mistakes appears to stem from issues with human attention. When presented with an image, annotators often miss obvious attributes, such as a cat is furry, and they also miss rarely occurring attributes, such as a bottle being square. By changing the task slightly, we might be able to avoid this issue. For example, one task might present several images, and the annotator labels the same attribute for each image.

4.3. Polygon Labels

We considered two different polygon tasks, labeling objects and object parts, and got quite different results for each. Annotators seem to thoroughly enjoy labeling polygons around objects, and they generally did a very good job. We had to reject very few images, only 5%, and annotator comments were typically positive. In contrast, getting good part annotations was much more difficult.

Two common problems occur with the part annotation. First, the annotators only label a portion of the parts, which may simply be an issue of motivation or possibly due to the attention issue mentioned above. These can be resolved by breaking up the task. Another problem appears to be confusion about the meaning of part names, which appears to be a language issue. By only using annotators from the US, these problems decreased, but so did the overall throughput.

4.4. Masks

Obtaining segmentation masks for materials proved to be the most difficult of all the tasks. The requirements for a segmentation mask are higher than polygons, as they require near pixel accuracy. As with MSRC [13] and the Pascal Segmentation task [5], we are forced to designate areas in the mask where the value is unknown, typically along material or object boundaries. In our case, we erode the masks so that evaluation and learning are less likely to use incorrectly labeled pixels.

Furthermore, choosing the material labels posed some difficulty. Without detailed knowledge of an object instance, it is difficult to visually distinguish particular materials: Is a bottle made of glass or transparent plastic, is the body of a car plastic or painted metal, and what exactly are blimps made out of? This suggests that that if humans cannot make this distinction, then it is unreasonable to expect a computer to succeed.

Fortunately, we found that while the names may have caused confusion, the material masks did not span across multiple physical materials. This allowed us to easily adjust the names afterwards without changing the segmentation.

5. Quality Assurance

Many of the previously mentioned issues can be addressed by grading the results or only accepting results from trusted annotators. Grading is more robust, since it allows every instance to be inspected, at the cost of requiring more time or money. Establishing trust, through qualification tasks or a positive history, provides a cheaper way to identify good annotators, but is not completely reliable. Therefore, we find a trade-off between these two, giving us good results at lower costs.

Grading: Each of the previous annotation tasks fall into one of two classes where quality can be verified efficiently:

1. Multiple annotations can be compared easily: By comparing multiple annotations for the same object, inconsistencies can be identified quickly.

2. Quick visual inspection: The amount of time to visually inspect an annotation is significantly less than the time to perform the annotation itself.

For binary tasks, we collect labels for each object from multiple annotators, allowing us to resolve the disagreement with a majority vote, or flag it for manual inspection. This automatic comparison is best suited for binary annotations because visual inspection takes just as much time as the actual annotation. Furthermore, obtaining the labels is cheap and fast, and comparison is possible with simple boolean expressions.

In contrast, each of the localization tasks are time consuming and expensive to collect. Thus, collecting multiple annotations can be prohibitive. Also, comparing two polygons may be possible, but is not always reliable. Fortunately, visually verifying each of these annotations can be done at a glance, so grading is still feasible.

Establishing Trust: To reduce the number of potentially bad annotators and avoid inspecting every annotation, we can establish a certain level of trust in two ways. First, we can require them to pass a qualification task, which ensures that they have some grasp of the English language, and that they understand the task. Another consequence is that annotators looking for quick payment with little work are turned away by the perceived inconvenience of taking the qualification test. A second method to identify good annotators is to simply accept all work from an annotator that has submitted a certain number of tasks and demonstrated that they will be accepted with high probability. For labeling polygons, we accept everything from annotators that we have given an acceptance rate of 90% for 10 or more annotations for a particular task.

6. Results and Discussion

In this section, we give a qualitative and quantitative analysis of our annotation tasks. There are several common factors to consider when collecting data on mechanical turk. Quality, cost, and time are three interdependent measures for good annotations. Here, we spent little energy choosing an optimal pay rate, and time was of minimal concern, so we focused mainly on quality assurance. Sorokin and Forsyth [16] give a more detailed overview of the trade-off between these criteria.

To characterize these factors, we first present a general set of statistics for each task. Throughput, cost, pay rate and quality are the most representative criteria. Each of these can be found in Table 1.

6.1. ATT09

To provide a comparison with previous work on collecting semantic attributes, we compare to the ATT09 dataset. There are a total of 15980 objects from the Pascal and Yahoo datasets, with 81 unique attributes (64 of which are designated for experiments). The acceptance rate is only an estimate here, based on the agreement with experts on a subset of the data. This previous work uses a less stringent qualification process, and more importantly does not filter out international workers. This is reflected in the lower acceptance rate when compared to the CORE binary annotations.

6.2. CORE

To summarize the CORE dataset, we collect 2780 images, containing 3192 labeled objects from 28 categories. For these objects, we collect a total of 26695 polygons from one of 71 possible part types, and there are 34 possible binary attributes. Finally, there are 1052 images labeled with material masks, with 10 different material labels. Tables 2,3,4 give more details for the attributes and categories collected.

Quality Results: We found that the quality of annotation is heavily dependent on the task itself. For example, even with much lower pay rate, the object annotations have the highest acceptance rate. When considering the part polygon annotation, it seems that this task is too daunting for some annotators. The results from Figure 3 support this hypothesis, as we found that many of the bad annotations result from being completed too hastily. To resolve this, it may be helpful to split the task in half, and request annotations for a subset of the parts.

Out of curiosity, we performed a simple experiment to see how carefully annotators actually read the directions. For the part annotation task, we included a line at the end of the instructions asking the annotator to write in the comment box "I Understand the Directions" with the first submitted task. Only 20.47% of the annotators actually followed these instructions. Unfortunately, this was not a helpful indicator of annotator reliability.

Suitable Tasks: This leads to a discussion of which tasks are more suitable for Mechanical Turk, and what expectations to set for these tasks. First, getting pixel perfect annotations either requires paying a small pool of qualified works a high rate per task, or rejecting a large number of tasks. Therefore, for these sorts of tasks, Mechanical Turk may not be the best solution. The average worker is however willing to provide moderately accurate results, as characterized by column 2 of Figure 2, quickly and at a low cost.

Furthermore, tasks with a small language dependence are appealing for Mechanical Turk, because many of the

Task Class	Task Type	Time/Task (s)	Cost/Task ($)	Pay Rate ($/h)	Accept Rate (%)	Quantity	Cost ($)
Binary	ATT09	42	0.03	2.59	81.4	15980	559.30
	CORE	73	0.04	1.98	90.7	9576	430.92
Polygon	Object	160	0.03	0.67	95.0	2780	97.30
	Part	286	0.10	1.26	75.4	3192	351.12
Mask	Material	372	0.07	0.68	78.3	1052	81.00

Table 1. Summary of statistics for tasks: This table shows that some tasks are more difficult than others. Although the pay rate for part polygons is twice as high as object polygons, the quality is significantly lower. The difference in times between binary attributes for ATT09 and CORE can be explained by a more stringent qualification process, and is reflected in the acceptance rates.Total cost takes into account Mechanical Turk commissions.

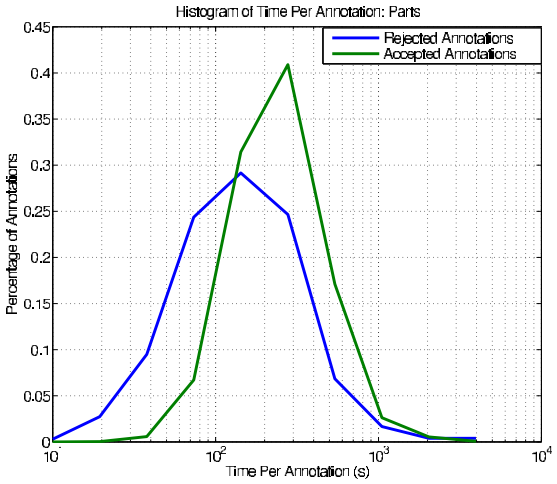


Figure 3. This graph shows a histogram of the amount of time spent on good and bad object part annotations. It is clear that many annotators did not want to spend sufficient time to produce high quality results. This may be an indication that higher pay could produce better results. This disparity in annotation time is not as prominent for the other tasks.

Object Polygons					
Name	#Obj	#parts	Name	#Obj	#parts
airplane	104	9.49	dolphin	151	6.17
alligator	122	8.90	eagle	107	8.01
bat	121	8.55	elephant	117	11.97
bicycle	103	6.62	elk	112	11.47
blimp	100	5.29	hovercraft	105	4.81
boat	110	3.76	jetski	110	3.64
bus	113	11.32	lizard	110	9.27
camel	129	12.15	monkey	117	11.90
car	110	8.15	motorcycle	87	9.03
carriage	105	5.43	penguin	151	6.95
cat	104	11.50	semi	110	11.51
cow	146	10.93	ship	105	4.29
crow	112	8.08	snowmobile	133	5.99
dog	103	13.17	whale	95	4.82

Table 2. All 28 Objects in the CORE dataset. For each category we list the number of objects (#Obj) and the average number of parts annotated for each instance (#parts). Each category has a rich list of parts, although some categories such as jetskis and ships have smaller lists of distinctive parts.

annotators are not native English speakers. Unfortunately, for many of our tasks, we are interested in the explicit semantics related to objects, implying a language dependence. The only other solution to leverage the potentially untapped workers is to provide the tasks in a variety of languages.

7. Conclusion

We have presented a new dataset that provides rich descriptions of object to encourage new areas of research in object representations and recognition. After resolving many of the difficulties of collecting these detailed annotations, we have the tools in place to allow us to expand the dataset to new domains and categories, allowing the capabilities of recognition systems to expand as well.

With this case study, we have also identified some of the shortcomings and limitations of Mechanical Turk, which is a valuable tool that is often tricky to tame.

8. Acknowledgements

This work was supported in part by a Google Research Award and the National Science Foundation under IIS-0904209, 0534837, and 0803603, and in part by the Office of Naval Research under N00014-01-1-0890 as part of the MURI program. Ali Farhadi was supported by Google Ph.D fellowship. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of NSF, Google, or ONR. The authors would like to thank Alexander Sorokin for insightful discussions on Mechanical Turk.

References

[1] Amazon mechanical turk. https://www.mturk.com. 1

[2] Flickr-photo sharing. http://www.flickr.com. 3

Figure 2. Annotation Results. Each row gives results for the object, part, and material tasks, from top to bottom. From left to right, we characterize the results as good, acceptable, and rejected. For *objects*, we want a polygon for each object, and the polygons to be tight around the object. For *object parts*, we also want every part to be labeled, and these parts to be tight. For the rejected example, the annotator missed several parts and annotated part of another plane. For *materials*, the mask should cover all relevant portions of the object, while respecting material boundaries. The rejected example demonstrates a mask that flows over boundaries, but more importantly, commonly occurring language deficiencies.

[3] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *ICCV*, 2009. 2

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 1, 2

[5] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results. http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html. 4

[6] A. Farhadi, I. Endres, and D. Hoiem. Attribute-centric recognition for cross-category generalization. In *CVPR*, 2010. 2

[7] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *CVPR*, 2009. 2

[8] R. Fergus, P. Perona, and A. Zisserman. Weakly supervised scale-invariant learning of models for visual recognition. *IJCV*, 71(3):273–303, 2007. 1, 2

[9] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and Simile Classifiers for Face Verification. In *IEEE International Conference on Computer Vision (ICCV)*, Oct 2009. 1, 2

[10] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, 2009. 1

[11] J. Luo, B. Caputo, and V. Ferrari. Who.s doing what: Joint modeling of names and verbs for simultaneous face and pose annotation. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta,

Binary Attributes			
Name	#Objs	Name	#Objs
another object	1047	P: jumping	6
C: above water	77	P: lying	222
C: docked in water	66	P: perched	154
C: flying	131	P: upright	19
C: in air	27	P: upside down	63
C: in water	269	P: running	0
C: on land	297	P: sitting	164
C: underwater	122	P: standing	509
P: driver	74	P: walking	140
P: wings extend	157	P: horse drawn	89
A: Back	625	P: pulling	1
A: Bottom	252	P: rider	400
A: Front	1341	C: snow	114
A: Left side	1224	unusual	118
A: Right side	1216	too small	31
A: Top	610	typical	2644
P: flying	84	not occluded	2721
P: hanging	9		

Table 3. The CORE dataset contains 35 binary attributes of several types. C: the context indicates the relation of the object to the scene. P: pose indicates the canonical name for the arrangement of parts and relation to other objects. A: aspect indicates the orientation of the object with respect to the camera, indicated by which faces of the object are visible.

editors, *Advances in Neural Information Processing Systems 22*, pages 1168–1176. 2009. 1

[12] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. LabelMe: a database and web-based tool for image annotation. Technical report, MIT, 2005. 1

[13] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, 2006. 4

[14] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering objects and their localization in images. *Computer Vision, IEEE International Conference on*, 1:370–377, 2005. 1

[15] A. Sorokin. Vision at large: large scale data collection for computer vision research. http://vision.cs.uiuc.edu/annotation/, 2010. 2

[16] A. Sorokin and D. Forsyth. Utility data annotation with amazon mechanical turk. 2008. 5

[17] Z. Yao, X. Yang, and S. Zhu. Introduction to a large scale general purpose groundtruth dataset: methodology, annotation tool, and benchmarks. *EMMCVPR*, 2007. 1

Part Polygons					
Name	#Cat	#Objs	Name	#Cat	#Objs
air cushon	1	99	light	3	38
arm	1	95	mane	1	37
balloon	1	100	mast	1	36
basket	1	26	mouth	15	873
beak	3	285	nose	13	779
cabin	5	270	oar	1	8
cargo	1	32	pedal	1	85
door	5	182	propeller	3	192
dorsal fin	2	136	rear win	1	2
ear	8	776	rear win	3	48
engine	3	139	reflector	1	23
exhaust	1	36	rein	3	19
exh pipe	7	206	row of win	5	252
eye	15	1332	saddle	2	40
fender	2	80	sail	1	20
fin	1	99	seat	6	428
flipper	2	180	side mir	7	331
foot	13	905	side win	4	279
front win	1	5	skis	1	110
front win	3	243	snout	12	582
gas tank	1	44	stop sign	1	8
gear	1	41	tail	15	1061
hand	1	85	tail fin	2	117
handlebar	4	357	tail light	4	135
harness	1	41	tail wing	1	89
head	15	1618	torso	15	1566
headlight	4	293	track	1	59
headlights	1	91	trailer	1	82
horn	2	109	trunk	1	91
hull	3	309	tusk	1	41
hump	1	96	veh wing	1	103
jet engine	1	52	wheel	8	709
land gear	1	38	window	4	136
leg	13	1235	windsh	3	189
lic plate	4	236	wing	4	438
lifeboat	1	29			

Table 4. There are 71 parts in CORE dataset. We also include a list of the number of categories that have this part (#Cat) and the number of objects that are assigned this part (#Objs).