Int J Comput Vis (2008) 77: 157–173
DOI 10.1007/s11263-007-0090-8

LabelMe: A Database and Web-Based Tool for Image Annotation

Bryan C. Russell · Antonio Torralba · Kevin P. Murphy · William T. Freeman

Received: 6 September 2005 / Accepted: 11 September 2007 / Published online: 31 October 2007

Abstract We seek to build a large collection of images with ground truth labels to be used for object detection and recognition research. Such data is useful for supervised learning and quantitative evaluation. To achieve this, we developed a web-based tool that allows easy image annotation and instant sharing of such annotations. Using this annotation tool, we have collected a large dataset that spans many object categories, often containing multiple instances over a wide variety of images. We quantify the contents of the dataset and compare against existing state of the art datasets used for object recognition and detection. Also, we show how to extend the dataset to automatically enhance object labels with WordNet, discover object parts, recover a depth ordering of objects in a scene, and increase the number of labels using minimal user supervision and images from the web.

Keywords Database · Annotation tool · Object recognition · Object detection

The first two authors (B.C. Russell and A. Torralba) contributed equally to this work.

B.C. Russell (✉) · A. Torralba · W.T. Freeman
Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139, USA
e-mail: brussell@csail.mit.edu

A. Torralba
e-mail: torralba@csail.mit.edu

W.T. Freeman
e-mail: billf@csail.mit.edu

K.P. Murphy
Departments of computer science and statistics, University of British Columbia, Vancouver, BC V6T 1Z4, Canada
e-mail: murphyk@cs.ubc.ca

1 Introduction

Thousands of objects occupy the visual world in which we live. Biederman (1987) estimates that humans can recognize about 30 000 entry-level object categories. Recent work in computer vision has shown impressive results for the detection and recognition of a few different object categories (Viola and Jones 2001; Heisele et al. 2001; Leibe and Schiele 2003). However, the size and contents of existing datasets, among other factors, limit current methods from scaling to thousands of object categories. Research in object detection and recognition would benefit from large image and video collections with ground truth labels spanning many different object categories in cluttered scenes. For each object present in an image, the labels should provide information about the object's identity, shape, location, and possibly other attributes such as pose.

By analogy with the speech and language communities, history has shown that performance increases dramatically when more labeled training data is made available. One can argue that this is a limitation of current learning techniques, resulting in the recent interest in Bayesian approaches to learning (Fei-Fei et al. 2003; Sudderth et al. 2005b) and multi-task learning (Torralba et al. 2004). Nevertheless, even if we can learn each class from just a small number of examples, there are still many classes to learn.

Large image datasets with ground truth labels are useful for supervised learning of object categories. Many algorithms have been developed for image datasets where all training examples have the object of interest well-aligned with the other examples (Turk and Pentland 1991; Heisele et al. 2001; Viola and Jones 2001). Algorithms that exploit context for object recognition (Torralba 2003; Hoiem et al. 2006) would benefit from datasets with many labeled object classes embedded in complex scenes. Such datasets should

contain a wide variety of environments with annotated objects that co-occur in the same images.

When comparing different algorithms for object detection and recognition, labeled data is necessary to quantitatively measure their performance (the issue of comparing object detection algorithms is beyond the scope of this paper; see Agarwal et al. (2004), Leibe (2005) for relevant issues). Even algorithms requiring no supervision (Sivic et al. 2005; Russell et al. 2006; Fei-Fei et al. 2003; Sudderth et al. 2005b, 2005a; Quelhas et al. 2005) need this quantitative framework.

Building a large dataset of annotated images with many objects is a costly and lengthy enterprise. Traditionally, datasets are built by a single research group and are tailored to solve a specific problem. Therefore, many currently available datasets only contain a small number of classes, such as faces, pedestrians, and cars. Notable exceptions are the Caltech 101 dataset (Fei-Fei et al. 2004), with 101 object classes (this was recently extended to 256 object classes Griffin et al. 2006), the PASCAL collection (Everingham et al. 2005), and the CBCL-streetscenes database (Bileschi 2006).

We wish to collect a large dataset of annotated images. To achieve this, we consider web-based data collection methods. Web-based annotation tools provide a way of building large annotated datasets by relying on the collaborative effort of a large population of users (http://www.flickr.com, von Ahn and Dabbish 2004; Russell et al. 2005; Stork 1999). Recently, such efforts have had much success. The Open Mind Initiative (Stork 1999) aims to collect large datasets from web users so that intelligent algorithms can be developed. More specifically, common sense facts are recorded (e.g. red is a primary color), with over 700K facts recorded to date. This project is seeking to extend their dataset with speech and handwriting data. Flickr (http://www.flickr.com) is a commercial effort to provide an online image storage and organization service. Users often provide textual tags to provide a caption of depicted objects in an image. Another way lots of data has been collected is through an online game that is played by many users. The ESP game (von Ahn and Dabbish 2004) pairs two random online users who view the same target image. The goal is for them to try to "read each other's mind" and agree on an appropriate name for the target image as quickly as possible. This effort has collected over 10 million image captions since 2003, with the images randomly drawn from the web. While the amount of data collected is impressive, only caption data is acquired. Another game, Peekaboom (von Ahn et al. 2006) has been created to provide location information of objects. While location information is provided for a large number of images, often only small discriminant regions are labeled and not entire object outlines.

In this paper we describe LabelMe, a database and an online annotation tool that allows the sharing of images and annotations. The online tool provides functionalities such as drawing polygons, querying images, and browsing the database. In the first part of the paper we describe the annotation tool and dataset and provide an evaluation of the quality of the labeling. In the second part of the paper we present a set of extensions and applications of the dataset. In this section we see that a large collection of labeled data allows us to extract interesting information that was not directly provided during the annotation process. In the third part we compare the LabelMe dataset against other existing datasets commonly used for object detection and recognition.

2 LabelMe

In this section we describe the details of the annotation tool and the results of the online collection effort.

2.1 Goals of the LabelMe Project

There are a large number of publically available databases of visual objects (Torralba et al. 2004; Agarwal et al. 2004; Leibe et al. 2004; Opelt et al. 2006b; Everingham et al. 2006; Fei-Fei et al. 2004, 2007; Griffin et al. 2006; Carmichael and Hebert 2004; Li and Shapiro 2002; LeCun et al. 2004; Burianek et al. 2000). We do not have space to review them all here. However, we give a brief summary of the main features that distinguishes the LabelMe dataset from other datasets.

- Designed for object class recognition as opposed to instance recognition. To recognize an object class, one needs multiple images of different instances of the same class, as well as different viewing conditions. Many databases, however, only contain different instances in a canonical pose.
- Designed for learning about objects embedded in a scene. Many databases consist of small cropped images of object instances. These are suitable for training patch-based object detectors (such as sliding window classifiers), but cannot be used for training detectors that exploit contextual cues.
- High quality labeling. Many databases just provide captions, which specify that the object is present somewhere in the image. However, more detailed information, such as bounding boxes, polygons or segmentation masks, is tremendously helpful.
- Many diverse object classes. Many databases only contain a small number of classes, such as faces, pedestrians and cars (a notable exception is the Caltech 101 database, which we compare against in Sect. 4).

Fig. 1 A screenshot of the labeling tool in use. The user is shown an image along with possibly one or more existing annotations, which are drawn on the image. The user has the option of annotating a new object by clicking along the boundary of the desired object and indicating its identity, or editing an existing annotation. The user may annotate as many objects in the image as they wish

- Many diverse images. For many applications, it is useful to vary the scene type (e.g. nature, street, and office scenes), distances (e.g. landscape and close-up shots), degree of clutter, etc.
- Many non-copyrighted images. For the LabelMe database most of the images were taken by the authors of this paper using a variety of hand-held digital cameras. We also have many video sequences taken with a head-mounted web camera.
- Open and dynamic. The LabelMe database is designed to allow collected labels to be instantly shared via the web and to grow over time.

2.2 The LabelMe Web-Based Annotation Tool

The goal of the annotation tool is to provide a drawing interface that works on many platforms, is easy to use, and allows instant sharing of the collected data. To achieve this, we designed a Javascript drawing tool, as shown in Fig. 1. When the user enters the page, an image is displayed. The image comes from a large image database covering a wide range of environments and several hundred object categories. The user may label a new object by clicking control points along the object's boundary. The user finishes by clicking on the starting control point. Upon completion, a popup dialog bubble will appear querying for the object name. The user freely types in the object name and presses enter to close the bubble. This label is recorded on the LabelMe server and is displayed on the presented image. The label is immediately available for download and is viewable by subsequent users who visit the same image.

The user is free to label as many objects depicted in the image as they choose. When they are satisfied with the number of objects labeled in an image, they may proceed to label another image from a desired set or press the *Show Next Image* button to see a randomly chosen image. Often, when a user enters the page, labels will already appear on the image. These are previously entered labels by other users. If there is a mistake in the labeling (either the outline or text label is not correct), the user may either edit the label by renaming the object or delete and redraw along the object's boundary. Users may get credit for the objects that they label by entering a username during their labeling session. This is recorded with the labels that they provide. The resulting labels are stored in the XML file format, which makes the annotations portable and easy to extend.

The annotation tool design choices emphasizes simplicity and ease of use. However, there are many concerns with this annotation collection scheme. One important concern is quality control. Currently quality control is provided by the users themselves, as outlined above. Another issue is the complexity of the polygons provided by the users (i.e. do users provide simple or complex polygon boundaries?). Another issue is what to label. For example, should one label the entire body, just the head, or just the face of a pedestrian? What if it is a crowd of people? Should all of the people be labeled? We leave these decisions up to each user. In this way, we hope the annotations will reflect what various people think are natural ways of segmenting an image. Finally, there is the text label itself. For example, should the object be labeled as a "person", "pedestrian", or "man/woman"? An obvious solution is to provide a drop-down menu of standard object category names. However, we prefer to let people use their own descriptions since these may capture some nuances that will be useful in the future. In Sect. 3.1, we de-

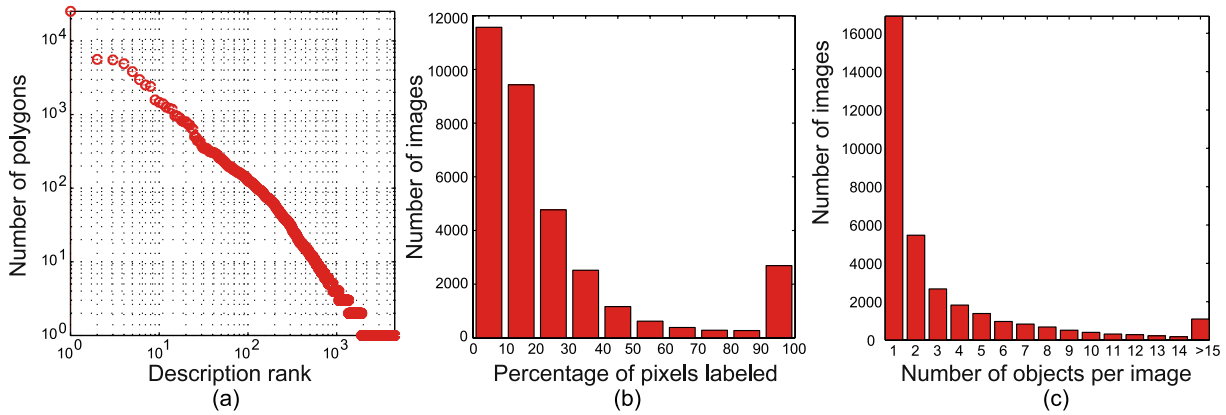


Fig. 2 Summary of the database content. **a** Sorted histogram of the number of instances of each object description. Notice that there is a large degree of consensus with respect to the entered descriptions. **b** Histogram of the number of annotated images as a function of the area labeled. The first bin shows that 11 571 images have less than 10% of the pixels labeled. The last bin shows that there are 2690 pictures with more than 90% of the pixels labeled. **c** Histogram of the number of labeled objects per image

scribe how to cope with the text label variability via WordNet (Fellbaum 1998). All of the above issues are revisited, addressed, and quantified in the remaining sections.

A Matlab toolbox has been developed to manipulate the dataset and view its contents. Example functionalities that are implemented in the toolbox allow dataset queries, communication with the online tool (this communication can in fact allow one to only download desired parts of the dataset), image manipulations, and other dataset extensions (see Sect. 3).

The images and annotations are organized online into folders, with the folder names providing information about the image contents and location of the depicted scenes/objects. The folders are grouped into two main categories: static pictures and sequences extracted from video. Note that the frames from the video sequences are treated as independent static pictures and that ensuring temporally consistent labeling of video sequences is beyond the scope of this paper. Most of the images have been taken by the authors using a variety of digital cameras. A small proportion of the images are contributions from users of the database or come from the web. The annotations come from two different sources: the LabelMe online annotation tool and annotation tools developed by other research groups. We indicate the sources of the images and annotations in the folder name and in the XML annotation files. For all statistical analyses that appear in the remaining sections, we will specify which subset of the database subset was used.

2.3 Content and Evolution of the LabelMe Database

We summarize the content of the LabelMe database as of December 21, 2006. The database consists of 111 490 polygons, with 44 059 polygons annotated using the online tool and 67 431 polygons annotated offline. There are 11 845 static pictures and 18 524 sequence frames with at least one object labeled.

As outlined above, a LabelMe description corresponds to the raw string entered by the user to define each object. Despite the lack of constraint on the descriptions, there is a large degree of consensus. Online labelers entered 2888 different descriptions for the 44 059 polygons (there are a total of 4210 different descriptions when considering the entire dataset). Figure 2a shows a sorted histogram of the number of instances of each object description for all 111 490 polygons.[1] Notice that there are many object descriptions with a large number of instances. While there is much agreement among the entered descriptions, object categories are nonetheless fragmented due to plurals, synonyms, and description resolution (e.g. “car”, “car occluded”, and “car side” all refer to the same category). In Sect. 3.1 we will address the issue of unifying the terminology to properly index the dataset according to real object categories.

Figure 2b shows a histogram of the number of annotated images as a function of the percentage of pixels labeled per image. The graph shows that 11 571 pictures have less than 10% of the pixels labeled and around 2690 pictures have more than 90% of labeled pixels. There are 4258 images with at least 50% of the pixels labeled. Figure 2c shows a histogram of the number of images as a function of the number of objects in the image. There are, on average, 3.3 annotated objects per image over the entire dataset. There

[1] A partial list of the most common descriptions for all 111 490 polygons in the LabelMe dataset, with counts in parenthesis: person walking (25 330), car (6548), head (5599), tree (4909), window (3823), building (2516), sky (2403), chair (1499), road (1399), bookshelf (1338), trees (1260), sidewalk (1217), cabinet (1183), sign (964), keyboard (949), table (899), mountain (823), car occluded (804), door (741), tree trunk (718), desk (656).

Fig. 3 Examples of annotated scenes. These images have more than 80% of their pixels labeled and span multiple scene categories. Notice that many different object classes are labeled per image

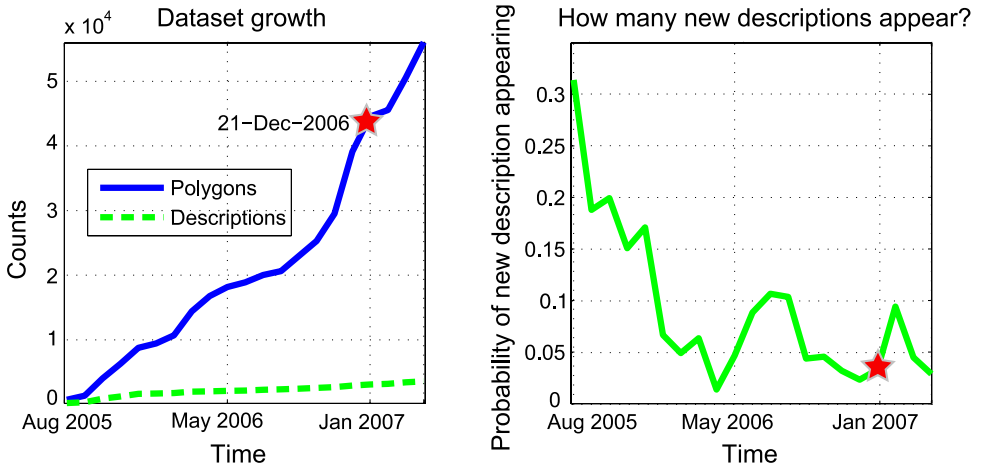


Fig. 4 Evolution of the online annotation collection over time. *Left*: total number of polygons (*blue, solid line*) and descriptions (*green, dashed line*) in the LabelMe dataset as a function of time. *Right*: the probability of a new description being entered into the dataset as a function of time. Note that the graph plots the evolution through March 23rd, 2007 but the analysis in this paper corresponds to the state of the dataset as of December 21, 2006, as indicated by the *star*. Notice that the dataset has steadily increased while the rate of new descriptions entered has decreased

are 6876 images with at least 5 objects annotated. Figure 3 shows images depicting a range of scene categories, with the labeled objects colored to match the extent of the recorded polygon. For many images, a large number of objects are labeled, often spanning the entire image.

The web-tool allows the dataset to continuously grow over time. Figure 4 depicts the evolution of the dataset since the annotation tool went online. We show the number of new polygons and text descriptions entered as a function of time. For this analysis, we only consider the 44 059 polygons entered using the web-based tool. The number of new polygons increased steadily while the number of new descriptions grew at a slower rate. To make the latter observation more explicit, we also show the probability of a new description appearing as a function of time (we analyze the raw text descriptions).

2.4 Quality of the Polygonal Boundaries

Figure 5 illustrates the range of variability in the quality of the polygons provided by different users for a few object categories. For the analysis in this section, we only use the 44 059 polygons provided online. For each object category, we sort the polygons according to the number of control points. Figure 5 shows polygons corresponding to the 25th, 50th, and 75th percentile with respect to the range of control points clicked for each category. Many objects can already be recognized from their silhouette using a small number of control points. Note that objects can vary with respect to the number of control points to indicate its boundary. For instance, a computer monitor can be perfectly described, in most cases, with just four control points. However, a detailed segmentation of a pedestrian might require 20 control points.

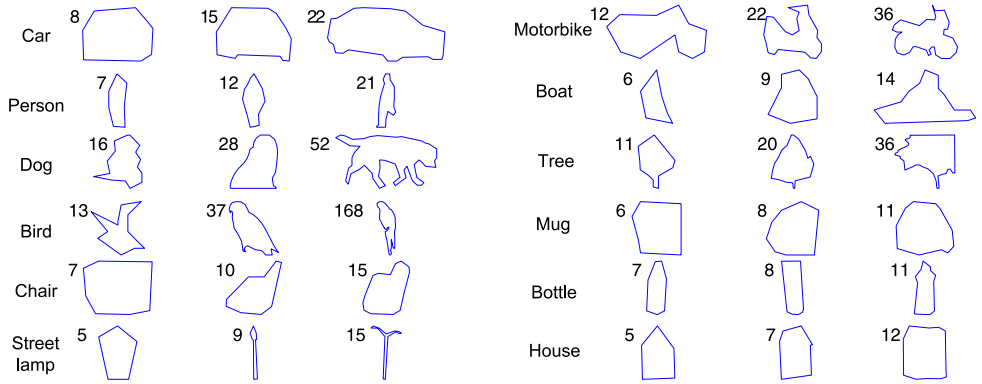


Fig. 5 Illustration of the quality of the annotations in the dataset. For each object we show three polygons depicting annotations corresponding to the 25th, 50th, and 75th percentile of the number of control points recorded for the object category. Therefore, the *middle* polygon corresponds to the average complexity of a segmented object class. The number of points recorded for a particular polygon appears near the *top-left* corner of each polygon. Notice that, in many cases, the object's identity can be deduced from its silhouette, often using a small number of control points

Fig. 6 Image crops of labeled objects and their corresponding silhouette, as given by the recorded polygonal annotation. Notice that, in many cases, the polygons closely follow the object boundary. Also, many diverse object categories are contained in the dataset

Figure 6 shows some examples of cropped images containing a labeled object and the corresponding recorded polygon.

2.5 Distributions of Object Location and Size

At first, one would expect objects to be uniformly distributed with respect to size and image location. For this to be true, the images should come from a photographer who randomly points their camera and ignores the scene. However, most of the images in the LabelMe dataset were taken by a human standing on the ground and pointing their camera towards interesting parts of a scene. This causes the location and size of the objects to not be uniformly distributed in the images. Figure 7 depicts, for a few object categories, a density plot showing where in the image each instance occurs and a histogram of object sizes, relative to the image size. Given how most pictures were taken, many of the cars can be found in the lower half region of the images. Note that for applications where it is important to have uniform prior distributions of object locations and sizes, we suggest cropping and rescaling each image randomly.

3 Extending the Dataset

We have shown that the LabelMe dataset contains a large number of annotated images, with many objects labeled per image. The objects are often carefully outlined using polygons instead of bounding boxes. These properties allow us to extract from the dataset additional information that was not provided directly during the labeling process. In this section we provide some examples of interesting extensions of the dataset that can be achieved with minimal user intervention. Code for these applications is available as part of the Matlab toolbox.

3.1 Enhancing Object Labels with WordNet

Since the annotation tool does not restrict the text labels for describing an object or region, there can be a large variance

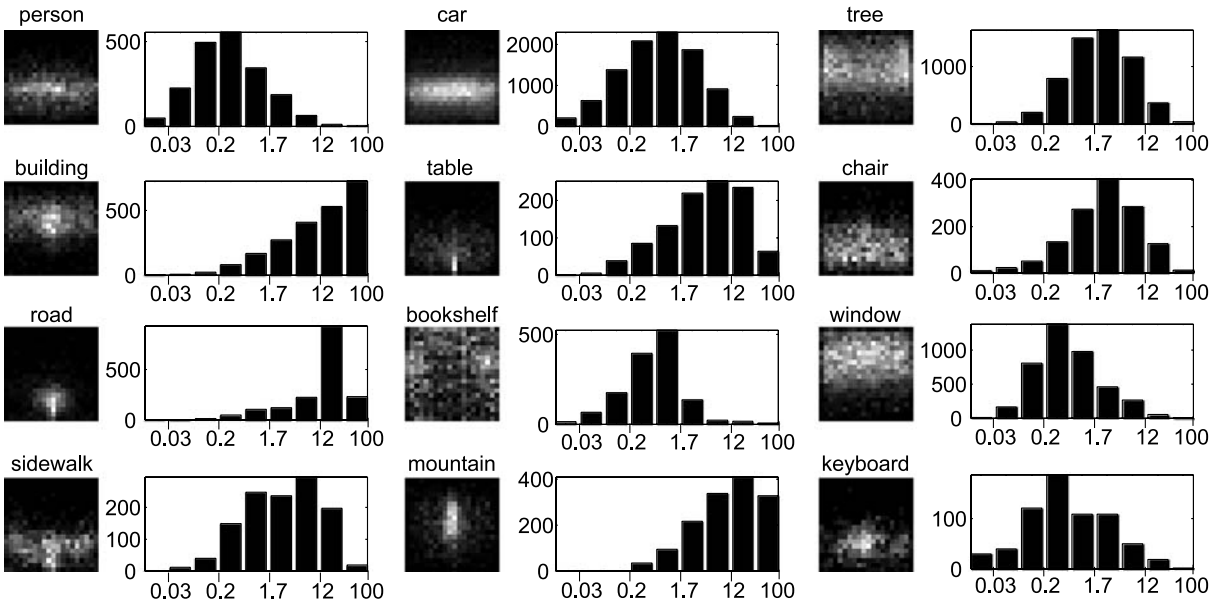


Fig. 7 Distributions of object location and size for a number of object categories in the LabelMe dataset. The distribution of locations are shown as a 2D histogram of the object centroid location in the different images (coordinates are normalized with respect to the image size). The size histogram illustrates what is the typical size that the object has in the LabelMe dataset. The *horizontal axis* is in logarithmic units and represents the percentage of the image area occupied by the object

of terms that describe the same object category. For example, a user may type any of the following to indicate the "car" object category: "car", "cars", "red car", "car frontal", "automobile", "suv", "taxi", etc. This makes analysis and retrieval of the labeled object categories more difficult since we have to know about synonyms and distinguish between object identity and its attributes. A second related problem is the level of description provided by the users. Users tend to provide basic-level labels for objects (e.g. "car", "person", "tree", "pizza"). While basic-level labels are useful, we would also like to extend the annotations to incorporate superordinate categories, such as "animal", "vehicle", and "furniture".

We use WordNet (Fellbaum 1998), an electronic dictionary, to extend the LabelMe descriptions. WordNet organizes semantic categories into a tree such that nodes appearing along a branch are ordered, with superordinate and subordinate categories appearing near the root and leaf nodes, respectively. The tree representation allows disambiguation of different senses of a word (polysemy) and relates different words with similar meanings (synonyms). For each word, WordNet returns multiple possible senses, depending on the location of the word in the tree. For instance, the word "mouse" returns four senses in WordNet, two of which are "computer mouse" and "rodent".[2] This raises the problem of sense disambiguation. Given a LabelMe description and multiple senses, we need to decide what the correct sense is.

WordNet can be used to automatically select the appropriate sense that should be assigned to each description (Ide and Vronis 1998). However, polysemy can prove challenging for automatic sense assignment. Polysemy can be resolved by analyzing the context (i.e. which other objects are present in the same image). To date, we have not found instances of polysemy in the LabelMe dataset (i.e. each description maps to a single sense). However, we found that automatic sense assignment produced too many errors. To avoid this, we allow for offline manual intervention to decide which senses correspond to each description. Since there are fewer descriptions than polygons (c.f. Fig. 4), the manual sense disambiguation can be done in a few hours for the entire dataset.

We extended the LabelMe annotations by manually creating associations between the different text descriptions and WordNet tree nodes. For each possible description, we queried WordNet to retrieve a set of senses, as described above. We then chose among the returned senses the one that best matched the description. Despite users entering text without any quality control, 3916 out of the 4210 (93%)

[2]The WordNet parents of these terms are (i) *computer mouse*: electronic device; device; instrumentality, instrumentation; artifact, artifact; whole, unit; object, physical object; physical entity; entity and (ii) *rodent*: rodent, gnawer, gnawing animal; placental, placental mammal, eutherian, eutherian mammal; mammal, mammalian; vertebrate, craniate; chordate; animal, animate being, beast, brute, creature, fauna; organism, being; living thing, animate thing; object, physical object; physical entity; entity.

Table 1 Examples of LabelMe descriptions returned when querying for the objects "person" and "car" after extending the labels with WordNet (not all of the descriptions are shown). For each description, the counts represents the number of returned objects that have the corresponding description. Note that some of the descriptions do not contain the query words

Person (27 719 polygons)		Car (10 137 polygons)	
Label	Polygon count	Label	Polygon count
Person walking	25 330	Car	6548
Person	942	Car occluded	804
Person standing	267	Car rear	584
Person occluded	207	Car side	514
Person sitting	120	Car crop	442
Pedestrian	121	Car frontal	169
Man	117	Taxi	8
Woman	75	Suv	4
Child	11	Cab	3
Girl	9	Automobile	2

unique LabelMe descriptions found a WordNet mapping, which corresponds to 104 740 out of the 111 490 polygon descriptions. The cost of manually specifying the associations is negligible compared to the cost of entering the polygons and must be updated periodically to include the newest descriptions. Note that it may not be necessary to frequently update these associations since the rate of new descriptions entered into LabelMe decreases over time (c.f. Fig. 4).

We show the benefit of adding WordNet to LabelMe to unify the descriptions provided by the different users. Table 1 shows examples of LabelMe descriptions that were returned when querying for "person" and "car" in the WordNet-enhanced framework. Notice that many of the original descriptions did not contain the queried word. Figure 8 shows how the number of polygons returned by one query (after extending the annotations with WordNet) are distributed across different LabelMe descriptions. It is interesting to observe that all of the queries seem to follow a similar law (linear in a log-log plot).

Table 2 shows the number of returned labels for several object queries before and after applying WordNet. In general, the number of returned labels increases after applying WordNet. For many specific object categories this increase is small, indicating the consistency with which that label is used. For superordinate categories, the number of returned matches increases dramatically. The object labels shown in Table 2 are representative of the most frequently occurring labels in the dataset.

One important benefit of including the WordNet hierarchy into LabelMe is that we can now query for objects at various levels of the WordNet tree. Figure 9 shows examples of queries for superordinate object categories. Very few of these examples were labeled with a description that matches the superordinate category, but nonetheless we can find them.

While WordNet handles most ambiguities in the dataset, errors may still occur when querying for object categories.

Table 2 Number of returned labels when querying the original descriptions entered into the labeling tool and the WordNet-enhanced descriptions. In general, the number of returned labels increases after applying WordNet. For entry-level object categories this increase is relatively small, indicating the consistency with which the corresponding description was used. In contrast, the increase is quite large for superordinate object categories. These descriptions are representative of the most frequently occurring descriptions in the dataset

Category	Original description	WordNet description
Person	27 019	27 719
Car	10 087	10 137
Tree	5997	7355
Chair	1572	2480
Building	2723	3573
Road	1687	2156
Bookshelf	1588	1763
Animal	44	887
Plant	339	8892
Food	11	277
Tool	0	90
Furniture	7	6957

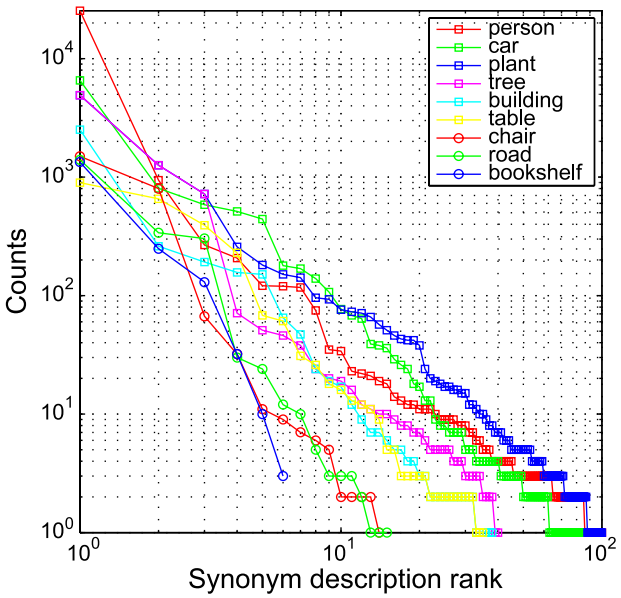


Fig. 8 How the polygons returned by one query (in the WordNet-enhanced framework) are distributed across different descriptions. The distributions seem to follow a similar law: a linear decay in a log-log plot with the number of polygons for each different description on the *vertical axis* and the descriptions (sorted by number of polygons) on the *horizontal axis*. Table 1 shows the actual descriptions for the queries "person" and "car"

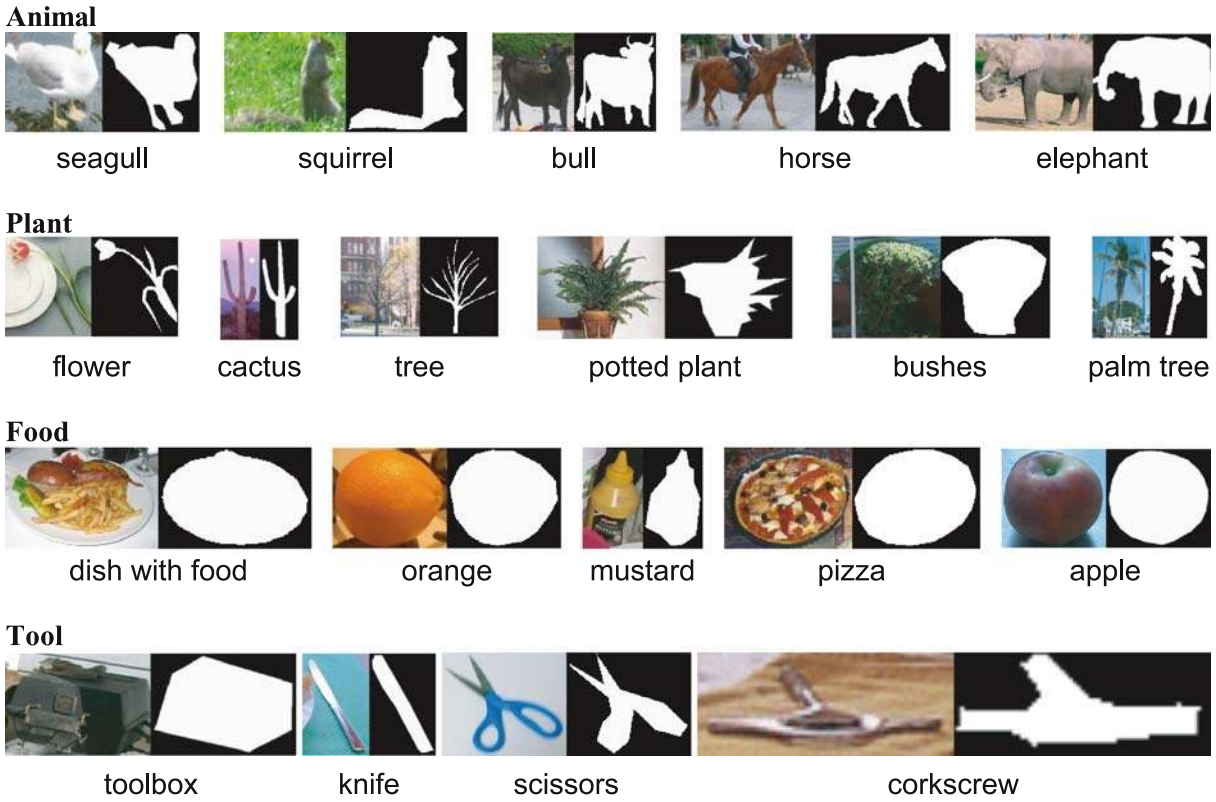


Fig. 9 Queries for superordinate object categories after incorporating WordNet. Very few of these examples were labeled with a description that matches the superordinate category (the original LabelMe descriptions are shown below each image). Nonetheless, we are able to retrieve these examples

The main source of error arises when text descriptions get mapped to an incorrect tree node. While this is not very common, it can be easily remedied by changing the text label to be more descriptive. This can also be used to clarify cases of polysemy, which our system does not yet account for.

3.2 Object-Parts Hierarchies

When two polygons have a high degree of overlap, this provides evidence of either (i) an object-part hierarchy or (ii) an occlusion. We investigate the former in this section and the latter in Sect. 3.3.

We propose the following heuristic to discover semantically meaningful object-part relationships. Let I_O denote the set of images containing a query object (e.g. car) and $I_P \subseteq I_O$ denote the set of images containing part P (e.g. wheel). Intuitively, for a label to be considered as a part, the label's polygons must consistently have a high degree of overlap with the polygons corresponding to the object of interest when they appear together in the same image. Let the overlap score between an object and part polygons be the ratio of the intersection area to the area of the part polygon. Ratios exceeding a threshold of 0.5 get classified as having high overlap. Let $I_{O,P} \subseteq I_P$ denote the images where object and part polygons have high overlap. The object-part score for a candidate label is $N_{O,P}/(N_P + \alpha)$ where $N_{O,P}$ and N_P are the number of images in $I_{O,P}$ and I_P respectively and α is a concentration parameter, set to 5. We can think of α as providing pseudocounts and allowing us to be robust to small sample sizes.

The above heuristic provides a list of candidate part labels and scores indicating how well they co-occur with a given object label. In general, the scores give good candidate parts and can easily be manually pruned for errors. Figure 10 shows examples of objects and proposed parts using the above heuristic. We can also take into account viewpoint information and find parts, as demonstrated for the car object category. Notice that the object-parts are semantically meaningful.

Once we have discovered candidate parts for a set of objects, we can assign specific part instances to their corresponding object. We do this using the intersection overlap heuristic, as above, and assign parts to objects where the intersection ratio exceeds the 0.5 threshold. For some robustness to occlusion, we compute a depth ordering of the polygons in the image (see Sect. 3.3) and assign the part to the polygon with smallest depth that exceeds the intersection ratio threshold. Figure 11 gives some quantitative re-

Fig. 10 Objects and their parts. Using polygon information alone, we automatically discover object-part relationships. We show example parts for the building, person, mountain, sky, and car object classes, arranged as constellations, with the object appearing in the center of its parts. For the car object class, we also show parts when viewpoint is considered

sults on the number of parts per object and the probability with which a particular object-part is labeled.

3.3 Depth Ordering

Frequently, an image will contain many partially overlapping polygons. This situation arises when users complete an occluded boundary or when labeling large regions containing small occluding objects. In these situations we need to know which polygon is on top in order to assign the image pixels to the correct object label. One solution is to request depth ordering information while an object is being labeled. Instead, we wish to reliably infer the relative depth ordering and avoid user input.

The problem of inferring depth ordering for overlapping regions is a simpler problem than segmentation. In this case we only need to infer who owns the region of intersection. We summarize a set of simple rules to decide the relative ordering of two overlapping polygons:

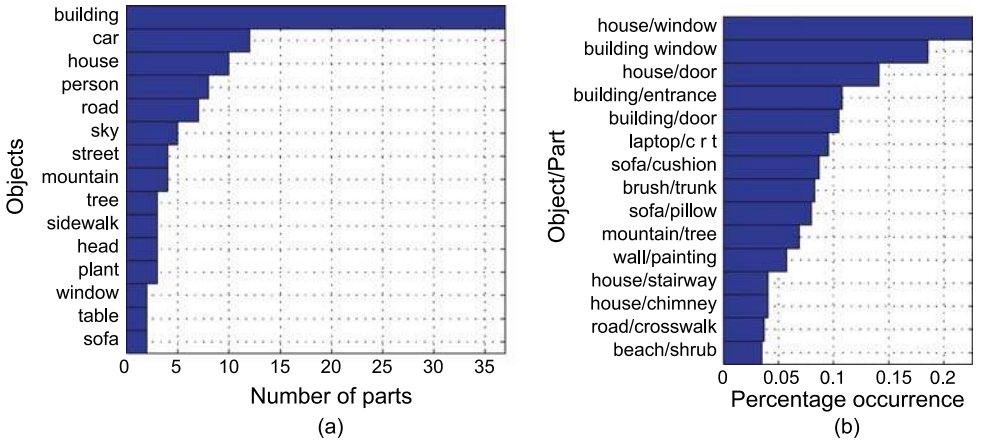


Fig. 11 Quantitative results showing **a** how many parts an object has and **b** the likelihood that a particular part is labeled when an object is labeled. Note that there are 29 objects with at least one discovered part (only 15 are shown here). We are able to discover a number of objects having parts in the dataset. Also, a part will often be labeled when an object is labeled

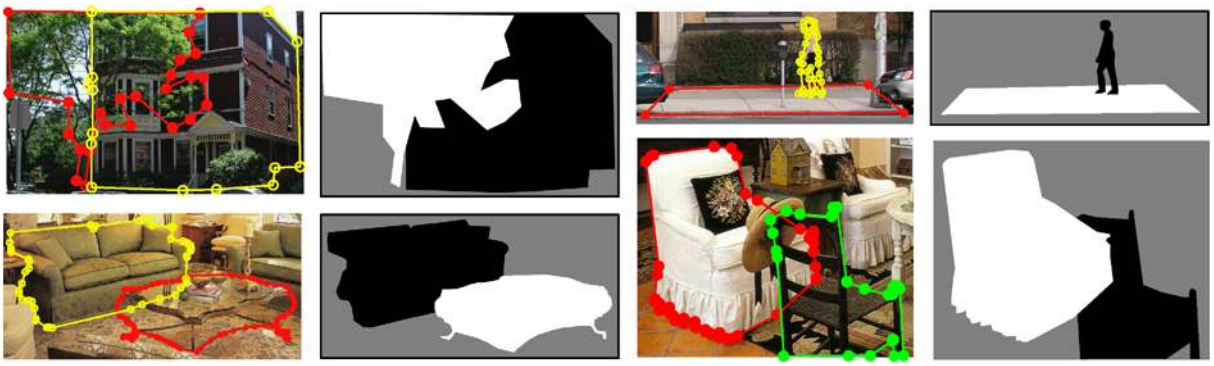


Fig. 12 Each image pair shows an example of two overlapping polygons and the final depth-ordered segmentation masks. Here, *white* and *black regions* indicate near and far layers, respectively. A set of rules (see text) were used to automatically discover the depth ordering of the overlapping polygon pairs. These rules provided correct assignments for 97% of 1000 polygon pairs tested. The *bottom right* example shows an instance where the heuristic fails. The heuristic sometimes fails for wiry or transparent objects

- Some objects are always on the bottom layer since they cannot occlude any objects. For instance, objects that do not own any boundaries (e.g. sky) and objects that are on the lowest layer (e.g. sidewalk and road).
- An object that is completely contained in another one is on top. Otherwise, the object would be invisible and, therefore, not labeled. Exceptions to this rule are transparent or wiry objects.
- If two polygons overlap, the polygon that has more control points in the region of intersection is more likely to be on top. To test this rule we hand-labeled 1000 overlapping polygon pairs randomly drawn from the dataset. This rule produced only 25 errors, with 31 polygon pairs having the same number of points within the region of intersection.
- We can also decide who owns the region of intersection by using image features. For instance, we can compute color histograms for each polygon and the region of intersection. Then, we can use histogram intersection (Swain and Ballard 1991) to assign the region of intersection to the polygon with the closest color histogram. This strategy achieved 76% correct assignments over the 1000 hand-labeled overlapping polygon pairs. We use this approach only when the previous rule could not be applied (i.e. both polygons have the same number of control points in the region of intersection).

Combining these heuristics resulted in 29 total errors out of the 1000 overlapping polygon pairs. Figure 12 shows some examples of overlapping polygons and the final assignments. The example at the bottom right corresponds to an error. In cases in which objects are wiry or transparent, the rule might fail. Figure 13 shows the final layers for scenes with multiple overlapping objects.

3.4 Semi-Automatic Labeling

Once there are enough annotations of a particular object class, one could train an algorithm to assist with the labeling. The algorithm would detect and segment additional instances in new images. Now, the user task would be to validate the detection (Vetter et al. 1997). A successful instance of this idea is the Seville project (Abramson and Freund 2005) where an incremental, boosting-based detector was trained. They started by training a coarse detector that was

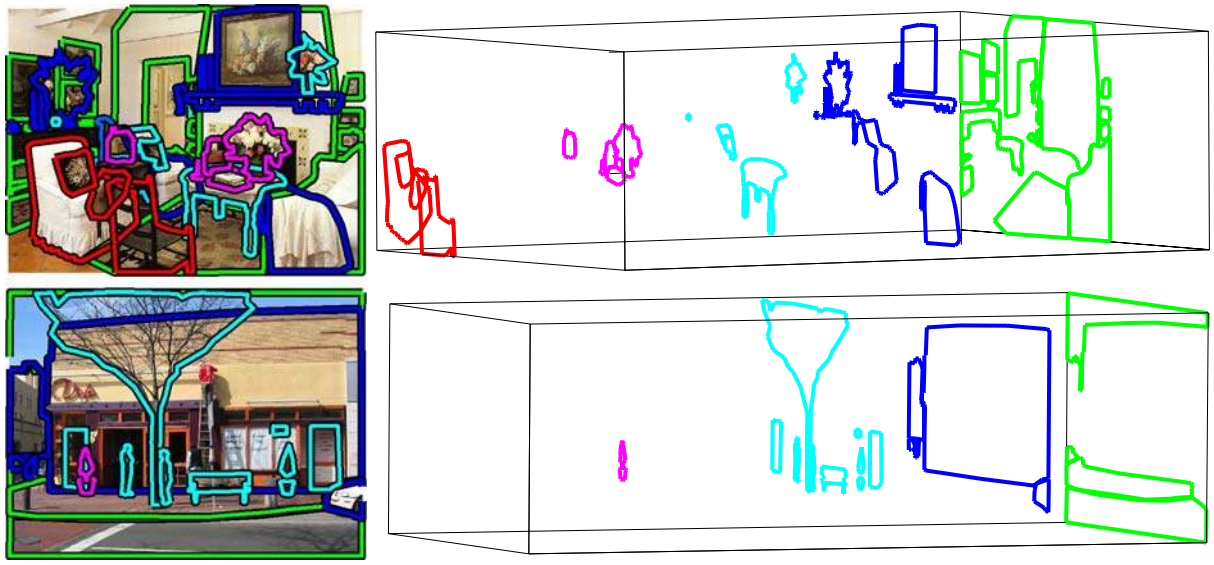


Fig. 13 Decomposition of a scene into layers given the automatic depth ordering recovery of polygon pairs. Since we only resolve the ambiguity between overlapping polygon pairs, the resulting ordering may not correspond to the real depth ordering of all the objects in the scene

good enough to simplify the collection of additional examples. The user provides feedback to the system by indicating when a bounding box was a correct detection or a false alarm. Then, the detector was trained again with the enlarged dataset. This process was repeated until a satisfactory number of images were labeled.

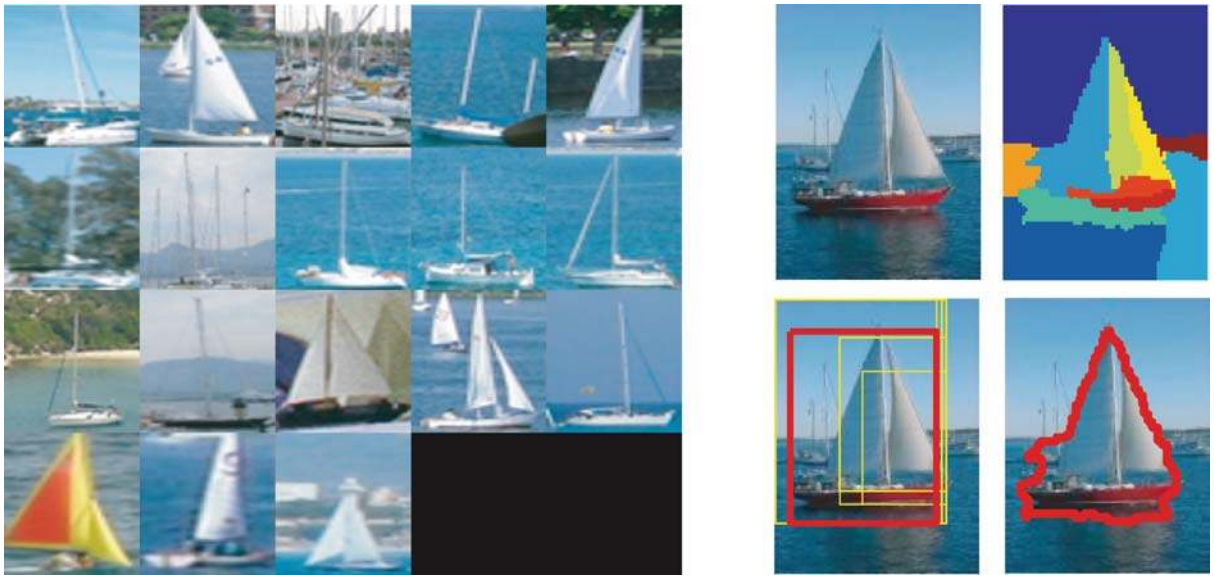
We can apply a similar procedure to LabelMe to train a coarse detector to be used to label images obtained from online image indexing tools. For instance, if we want more annotated samples of *sailboats*, we can query both LabelMe (18 segmented examples of sailboats were returned) and online image search engines (e.g. Google, Flickr, and Altavista). The online image search engines will return thousands of unlabeled images that are very likely to contain a sailboat as a prominent object. We can use LabelMe to train a detector and then run the detector on the retrieved unlabeled images. The user task will be to select the correct detections in order to expand the amount of labeled data.

Here, we propose a simple object detector. Although objects labeled with bounding boxes have proven to be very useful in computer vision, we would like the output of the automatic object detection procedure to provide polygonal boundaries following the object outline whenever possible.

- Find candidate regions: instead of running the standard sliding window, we propose creating candidate bounding boxes for objects by first segmenting the image to produce 10–20 regions. Bounding boxes are proposed by creating all the bounding boxes that correspond to combinations of these regions. Only the combinations that produce contiguous regions are considered. We also remove all candidate bounding boxes with aspect ratios outside the range defined by the training set. This results in a small set of candidates for each image (around 30 candidates).
- Compute features: resize each candidate region to a normalized size (96 × 96 pixels). Then, represent each candidate region with a set of features (e.g. bag of words Russell et al. 2006, edge fragments Opelt et al. 2006a, multiscale-oriented filters Oliva and Torralba 2001). For the experiments presented here, we used the Gist features (Oliva and Torralba 2001) (code available online) to represent each region.
- Perform classification: train a support vector machine classifier (Vapnik 1999) with a Gaussian kernel using the available LabelMe data and apply the classifier to each of the candidate bounding boxes extracted from each image. The output of the classifier will be a score for the bounding boxes. We then choose the bounding box with the maximum score and the segmentation corresponding to the segments that are inside the selected bounding box.

For the experiments presented here, we queried four object categories: sailboats, dogs, bottles, and motorbikes. Using LabelMe, we collected 18 sailboat, 41 dog, 154 bottle, and 49 motorbike images. We used these images to train four classifiers. Then, we downloaded 4000 images for each class from the web using Google, Flickr and Altavista. Not all of the images contained instances of the queried objects. It has been shown that image features can be used to improve the quality of the ranking returned by online queries (Fergus et al. 2005; Berg and Forsyth 2006). We used the detector trained with LabelMe to sort the images returned by the online query tools.

Figure 15 shows the results and compares the images sorted according to the ranking given by the output of the on-



(a) Sailboats from the LabelMe dataset (b) Detection and segmentation

Fig. 14 Using LabelMe to automatically detect and segment objects depicted in images returned from a web search. **a** Sailboats in the LabelMe dataset. These examples are used to train a classifier. **b** Detection and segmentation of a sailboat in an image download from the web using Google. First, we segment the image (*upper left*), which produces around 10 segmented regions (*upper right*). Then we create a list of candidate bounding boxes by combining all of the adjacent regions. Note that we discard bounding boxes whose aspect ratios lie outside the range of the LabelMe sailboat crops. Then we apply a classifier to each bounding box. We depict the bounding boxes with the highest scores (*lower left*), with the best scoring as a *thick bounding box* colored in *red*. The candidate segmentation is the outline of the regions inside the selected bounding box (*lower right*). After this process, a user may then select the correct detections to augment the dataset

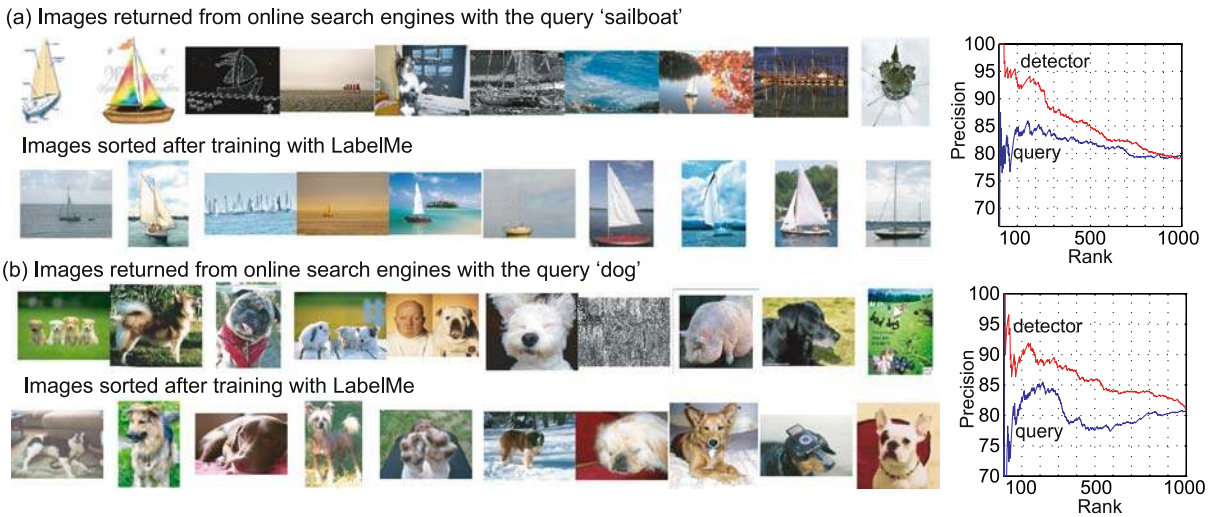


Fig. 15 Enhancing web-based image retrieval using labeled image data. Each pair of rows depict sets of sorted images for a desired object category. The *first row* in the pair is the ordering produced from an online image search using Google, Flickr and Altavista (the results of the three search engines are combined respecting the ranking of each image). The *second row* shows the images sorted according to the confidence score of the object detector trained with LabelMe. To better show how the performance decreases with rank, each row displays one out of every ten images. Notice that the trained classifier returns better candidate images for the object class. This is quantified in the graphs on the *right*, which show the precision (percentage correct) as a function of image rank

line search engines and the ranking provided by the score of the classifier. For each image we have two measures: (i) the rank in which the image was returned and (ii) the score of the classifier corresponding to the maximum score of all the candidate bounding boxes in the image. In order to measure performance, we provided ground truth for the first 1000 images downloaded from the web (for sailboats and dogs). The precision-recall graphs show that the score provided by the

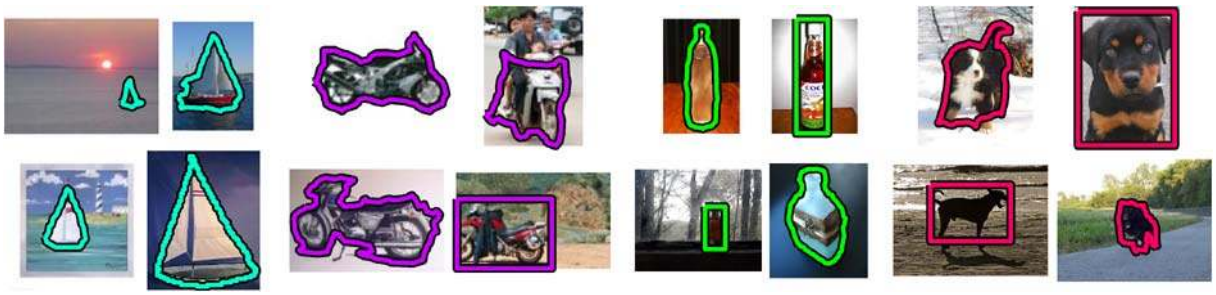


Fig. 16 Examples of automatically generated segmentations and bounding boxes for sailboats, motorbikes, bottles, and dogs

Table 3 Summary of datasets used for object detection and recognition research. For the LabelMe dataset, we provide the number of object classes with at least 30 annotated examples. All the other numbers provide the total counts

Dataset	# categories	# images	# annotations	Annotation type
LabelMe	183	30 369	111 490	Polygons
Caltech-101 (Fei-Fei et al. 2007)	101	8765	8765	Polygons
MSRC (Winn et al. 2005)	23	591	1751	Region masks
CBCL-Streetscenes (Bileschi 2006)	9	3547	27 666	Polygons
Pascal2006 (Everingham et al. 2006)	10	5304	5455	Bounding boxes

classifier provides a better measure of probability of presence of the queried object than the ranking in which the images are returned by the online tools. However, for the automatic labeling application, good quality labeling demands very good performance on the object localization task. For instance, in current object detection evaluations (Everingham et al. 2006), an object is considered correctly detected when the area of overlap between the ground truth bounding box and the detected bounding box is above 50% of the object size. However, this degree of overlap will not be considered satisfactory for labeling. Correct labeling requires above 90% overlap to be satisfactory.

After running the detectors on the 4000 images of each class collected from the web, we were able to select 162 sailboats, 64 dogs, 40 bottles, and 40 motorbikes that produced good annotations. This is shown in Fig. 16. The user had the choice to validate the segmentation or just the bounding box. The selection process is very efficient. Therefore, semi-automatic labeling may offer an interesting way of efficiently labeling images.

However, there are several drawbacks to this approach. First, we are interested in labeling full scenes with many objects, making the selection process less efficient. Second, in order for detection to work with a reasonable level of accuracy with current methods, the object needs to occupy a large portion of the image or be salient. Third, the annotated objects will be biased toward being easy to segment or detected. Note that despite semi-automatic labeling not being desirable for creating challenging benchmarks for evaluating object recognition algorithms, it can still be useful for training. There are also a number of applications that will benefit from having access to large amounts of labeled data, including image indexing tools (e.g. Flickr) and photorealistic computer graphics (Snavely et al. 2006). Therefore, creating semi-automatic algorithms to assist image labeling at the object level is an interesting area of application on its own.

4 Comparison with Existing Datasets for Object Detection and Recognition

We compare the LabelMe dataset against four annotated datasets currently used for object detection and recognition: Caltech-101 (Fei-Fei et al. 2007), MSRC (Winn et al. 2005), CBCL-Streetscenes (Bileschi 2006), and PASCAL2006 (Everingham et al. 2006). Table 3 summarizes these datasets. The Caltech-101 and CBCL-streetscenes provide location information for each object via polygonal boundaries. PASCAL2006 provides bounding boxes and MSRC provides segmentation masks.

For the following analysis with the LabelMe dataset, we only include images that have at least one object annotated and object classes with at least 30 annotated examples, resulting in a total of 183 object categories. We have also excluded, for the analysis of the LabelMe dataset, contributed annotations and sequences.

Figure 17a shows, for each dataset, the number of object categories and, on average, how many objects appear in an image. Notice that currently the LabelMe dataset contains more object categories than the existing datasets. Also, observe that the CBCL-Streetscenes and LabelMe datasets often have multiple annotations per image, indicating that the images correspond to scenes and contain multiple objects.

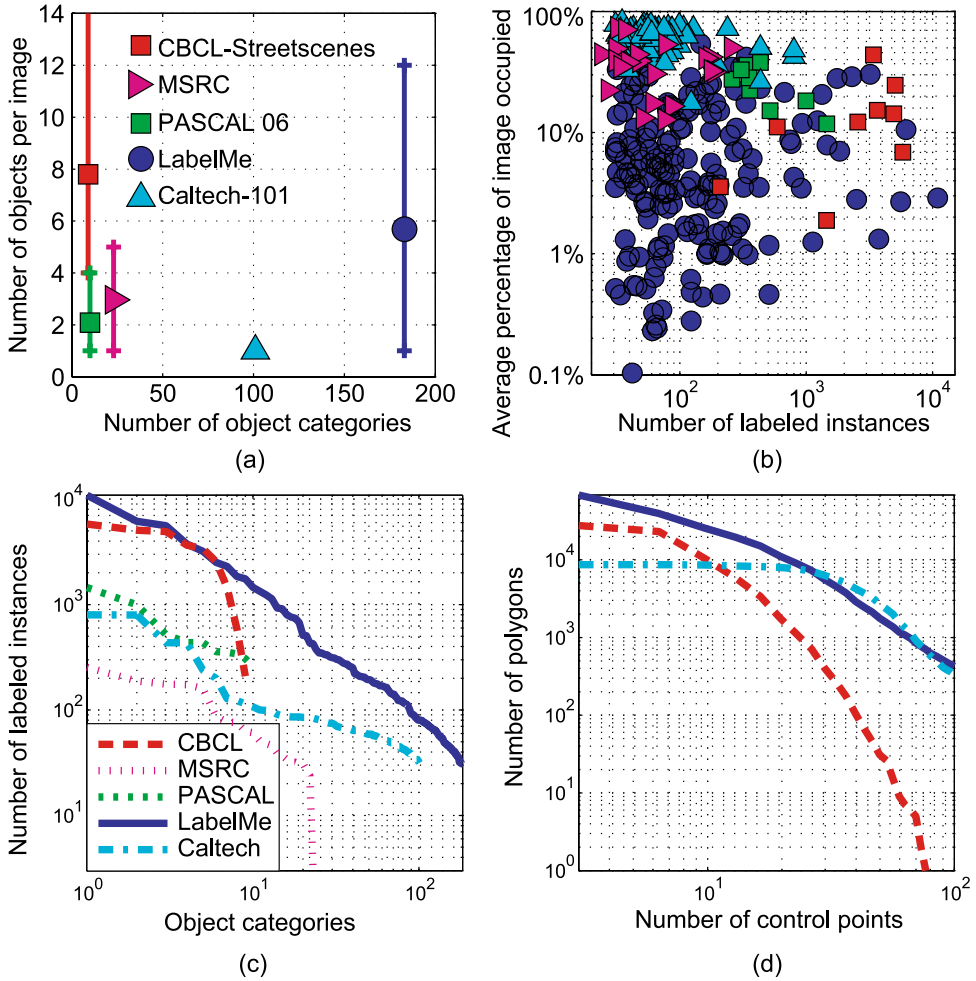


Fig. 17 Comparison of five datasets used for object detection and recognition: Caltech-101 (Fei-Fei et al. 2003), MSRC (Winn et al. 2005), CBCL-Streetscenes (Torralba et al. 2004), PASCAL2006 (Everingham et al. 2006), and LabelMe. **a** Number of object categories versus number of annotated objects per image. **b** Scatter plot of number of object category instances versus average annotation size relative to the image size, with each point corresponding to an object category. **c** Number of labeled instances per object category, sorted in decreasing order based on the number of labeled instances. Notice that the LabelMe dataset contains a large number of object categories, often with many instances per category, and has annotations that vary in size and number per image. This is in contrast to datasets prominently featuring one object category per image, making LabelMe a rich dataset and useful for tasks involving scene understanding. **d** Depiction of annotation quality, where the number of polygonal annotations are plotted as a function of the number of control points (we do not show the PASCAL2006 and MSRC datasets since their annotations correspond to bounding boxes and region masks, respectively)

This is in contrast with the other datasets, which prominently feature a small number of objects per image.

Figure 17b is a scatter plot where each point corresponds to an object category and shows the number of instances of each category and the average size, relative to the image. Notice that the LabelMe dataset has a large number of points, which are scattered across the entire plot while the other datasets have points clustered in a small region. This indicates the range of the LabelMe dataset: some object categories have a large number of examples (close to 10K examples) and occupy a small percentage of the image size. Contrast this with the other datasets where there are not as many examples per category and the objects tend to occupy a large portion of the image. Figure 17c shows the number of labeled instances per object category for the five datasets, sorted in decreasing order by the number of labeled instances. Notice that the line corresponding to the LabelMe dataset is higher than the other datasets, indicating the breadth and depth of the dataset.

We also wish to quantify the quality of the polygonal annotations. Figure 17d shows the number of polygonal annotations as a function of the number of control points. The LabelMe dataset has a wide range of control points and the number of annotations with many control points is large, indicating the quality of the dataset. The PASCAL2006 and MSRC datasets are not included in this analysis since their annotations consist of bounding boxes and region masks, respectively.

5 Conclusion

We described a web-based image annotation tool that was used to label the identity of objects and where they occur in images. We collected a large number of high quality annotations, spanning many different object categories, for a large set of images, many of which are high resolution. We presented quantitative results of the dataset contents showing the quality, breadth, and depth of the dataset. We showed how to enhance and improve the quality of the dataset through the application of WordNet, heuristics to recover object parts and depth ordering, and training of an object detector using the collected labels to increase the dataset size from images returned by online search engines. We finally compared against other existing state of the art datasets used for object detection and recognition.

Our goal is not to provide a new benchmark for computer vision. The goal of the LabelMe project is to provide a dynamic dataset that will lead to new research in the areas of object recognition and computer graphics, such as object recognition in context and photorealistic rendering.

Acknowledgements This work was supported by the National Science Foundation Grant No. 0413232, the National Geospatial-Intelligence Agency NEGI-1582-04-0004, the Office of Naval Research MURI Grant No. N00014-06-1-0734, the ARDA VACE program, the Canadian NSERC Discovery Grant program, and the Canadian Institute for Advanced Research.

References

Abramson, Y., & Freund, Y. (2005). Semi-automatic visual learning (seville): a tutorial on active learning for visual object recognition. In *International conference on computer vision and pattern recognition (CVPR'05)*, San Diego.

Agarwal, S., Awan, A., & Roth, D. (2004). Learning to detect objects in images via a sparse, part-based representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(11), 1475–1490.

Berg, T. L., & Forsyth, D. A. (2006). Animals on the web. In *CVPR* (Vol. 2, pp. 1463–1470).

Biederman, I. (1987). Recognition by components: a theory of human image interpretation. *Pyschological Review*, *94*, 115–147.

Bileschi, S. (2006). *CBCL streetscenes* (Technical report). MIT CBCL. The CBCL-Streetscenes dataset can be downloaded at http://cbcl.mit.edu/software-datasets.

Burianek, J., Ahmadyfard, A., & Kittler, J. (2000). Soil-47, the Surrey object image library. http://www.ee.surrey.ac.uk/Research/VSSP/demos/colour/soil47/.

Carmichael, O., & Hebert, M. (2004). Word: Wiry object recognition database. Carnegie Mellon University. www.cs.cmu.edu/~owenc/word.htm. Accessed January 2004.

Everingham, M., Zisserman, A., Williams, C., Van Gool, L., Allan, M., Bishop, C., Chapelle, O., Dalal, N., Deselaers, T., Dorko, G., Duffner, S., Eichhorn, J., Farquhar, J., Fritz, M., Garcia, C., Griffiths, T., Jurie, F., Keysers, D., Koskela, M., Laaksonen, J., Larlus, D., Leibe, B., Meng, H., Ney, H., Schiele, B., Schmid, C., Seemann, E., Shawe-Taylor, J., Storkey, A., Szedmak, S., Triggs, B., Ulusoy, I., Viitaniemi, V., & Zhang, J. (2005). The 2005 pascal visual object classes challenge. In *First PASCAL challenges workshop*. Springer.

Everingham, M., Zisserman, A., Williams, C. K. I., & Van Gool, L. (2006). *The pascal visual object classes challenge 2006 (voc 2006) results* (Technical report). September 2006. The PASCAL2006 dataset can be downloaded at http://www.pascal-network.org/challenges/VOC/voc2006/.

Fei-Fei, L., Fergus, R., & Perona, P. (2003). A bayesian approach to unsupervised one-shot learning of object categories. In *IEEE international conference on computer vision*.

Fei-Fei, L., Fergus, R., & Perona, P. (2004). Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In *IEEE CVPR 2004, workshop on generative-model based vision*.

Fei-Fei, L., Fergus, R., & Perona, P. (2007, in press). One-shot learning of object categories. *IEEE Transactions on Pattern Recognition and Machine Intelligence*. The Caltech 101 dataset can be downloaded at http://www.vision.caltech.edu/Image_Datasets/Caltech101/Caltech101.html.

Fellbaum, C. (1998). *Wordnet: An electronic lexical database*. Bradford Books.

Fergus, R., Fei-Fei, L., Perona, P., & Zisserman, A. (2005). Learning object categories from google's image search. In *Proceedings of the 10th international conference on computer vision* (Vol. 2, pp. 1816–1823). Beijing, China, October 2005.

Griffin, G., Holub, A. D., & Perona, P. (2006). *The Caltech-256* (Technical report). California Institute of Technology.

Heisele, B., Serre, T., Mukherjee, S., & Poggio, T. (2001). Feature reduction and hierarchy of classifiers for fast object detection in video images. In *CVPR*.

Hoiem, D., Efros, A., & Hebert, M. (2006). Putting objects in perspective. In *CVPR*.

Ide, N., & Vronis, J. (1998). Introduction to the special issue on word sense disambiguation: the state of the art. *Computational Linguistics*, *24*(1), 1–40.

LeCun, Y., Huang, F.-J., & Bottou, L. (2004). Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of CVPR'04*. Los Alamitos: IEEE Press.

Leibe, B. (2005). *Interleaved object categorization and segmentation*. Ph.D. thesis.

Leibe, B., & Schiele, B. (2003). Analyzing appearance and contour based methods for object categorization. In *IEEE conference on computer vision and pattern recognition (CVPR'03)*, Madison, WI, June 2003.

Leibe, B., Leonardis, A., & Schiele, B. (2004). Combined object categorization and segmentation with an implicit shape model. In *ECCV*.

Li, Y., & Shapiro, L. G. (2002). Consistent line clusters for building recognition in cbir. In *Proceedings of the international conference on pattern recognition*.

Oliva, A., & Torralba, A. (2001). Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal of Computer Vision*, *42*(3), 145–175.

Opelt, A., Pinz, A., & Zisserman, A. (2006a). A boundary-fragment-model for object detection. In *ECCV*.

Opelt, A., Pinz, A., Fussenegger, M., & Auer, P. (2006b). Generic object recognition with boosting. *IEEE Transactions on Pattern Recognition and Machine Intelligence (PAMI)*, *28*(3).

Quelhas, P., Monay, F., Odobez, J. M., Gatica-Perez, D., Tuytelaars, T., & Van Gool, L. (2005). Modeling scenes with local descriptors and latent aspects. In *IEEE international conference on computer vision*.

Russell, B. C., Torralba, A., Murphy, K. P., & Freeman, W. T. (2005). *Labelme: a database and web-based tool for image annotation* (Technical Report AIM-2005-025). MIT AI Lab Memo, September 2005.

Russell, B. C., Efros, A. A., Sivic, J., Freeman, W. T., & Zisserman, A. (2006). Using multiple segmentations to discover objects and their extent in image collections. In *CVPR*.

Sivic, J., Russell, B. C., Efros, A. A., Zisserman, A., & Freeman, W. T. (2005). Discovering objects and their location in images. In *IEEE international conference on computer vision*.

Snavely, N., Seitz, S. M., & Szeliski, R. (2006). Photo tourism: Exploring photo collections in 3d. *ACM Transactions on Graphics*, *25*(3), 137–154.

Stork, D. G. (1999). The open mind initiative. *IEEE Intelligent Systems and Their Applications*, *14*(3), 19–20.

Sudderth, E., Torralba, A., Freeman, W. T., & Willsky, W. (2005a). Describing visual scenes using transformed dirichlet processes. In *Advances in neural information processing systems*.

Sudderth, E., Torralba, A., Freeman, W. T., & Willsky, W. (2005b). Learning hierarchical models of scenes, objects, and parts. In *IEEE international conference on computer vision*.

Swain, M. J., & Ballard, D. H. (1991). Color indexing. *International Journal of Computer Vision*, *7*(1).

Torralba, A. (2003). Contextual priming for object detection. *International Journal of Computer Vision*, *53*(2), 153–167.

Torralba, A., Murphy, K., & Freeman, W. (2004). Sharing features: efficient boosting procedures for multiclass object detection. In *CVPR*.

Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, *3*(1), 71–86.

Vapnik, V. (1999). *The nature of statistical learning theory*. New York: Springer.

Vetter, T., Jones, M., & Poggio, T. (1997). A bootstrapping algorithm for learning linear models of object classes. In *CVPR*.

Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple classifiers. In *CVPR*.

von Ahn, L., & Dabbish, L. (2004). Labeling images with a computer game. In *Processing on SIGCHI conference on human factors in computing systems*.

von Ahn, L., Liu, R., & Blum, M. (2006). Peekaboom: A game for locating objects in images. In *In ACM CHI*.

Winn, J., Criminisi, A., & Minka, T. (2005). Object categorization by learned universal visual dictionary. In *IEEE international conference on computer vision*. The MSRC dataset can be downloaded at http://research.microsoft.com/vision/cambridge/recognition/default.htm.