Image Annotation & Segmentation Tool

A project report Submitted in partial fulfilment of requirement
for the award of the degree
Of
Bachelor of Technology
In
Computer Science and Engineering

Session 2019-20

Submitted by

Jay Prakash Sonkar (1604310029)
Mohammad Faizan (1704310907)
Vaibhav Singh (1604310054)
Rohit Kumar (1604310046)

Under the Supervision of

Dr. Yashpal Singh

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Bundelkhand Institute of Engineering and Technology,

Jhansi 284128

BUNDELKHAND INSTITUTE OF ENGINEERING AND TECHNOLOGY, JHANSI

Department of Computer Science and Engineering

2019-2020

CERTIFICATE

This is to certify that the project entitled **"Image Annotation & Segmentation Tool"** submitted by **Jay Prakash Sonkar (1604310029), Mohammad Faizan (1704310907), Vaibhav Singh (1604310054) and Rohit (1604310046)** in the partial fulfilment of the requirement for the award of Bachelor of Technology degree in Computer Science and Engineering at Bundelkhand Institute of Engineering and Technology, Jhansi is an authentic work carried out by them under my super supervision and guidance.

To the best of my knowledge, the matter embodied in the project has not been submitted to any other University/Institute for the award of any Degree or Diploma.

Project Guide:

…….

Dr. Yashpal Singh

Computer Science and Engineering

Date:

Place:

ACKNOWLEDGEMENT

We are deeply indebted to our respected Head of the Department **Prof. A.K. Solanki** for guiding us. The team is also grateful to our project guide **Dr. Yashpal Singh** for his indomitable contribution and guidance without which the completion of this project would have been impossible.

Our sincerest thanks to all our teachers **Prof. A.K. Solanki** , **Dr. Yashpal Singh** , **Dr. S.K Gupta , Dr. R.N. Verma , Dr. Mrityunjay Singh , Dr. S.P. Singh , Dr. Anuj Yadav , Er. Shivam Pandey ,** seniors and colleagues whose help and guidance brought this project to successful completion.

Group Members:

Jay Prakash Sonkar (1604310029)

Mohammad Faizan(1704310907)

Vaibhav Singh (1604310054)

Rohit Kumar (1604310046)

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 GUI-based tool that allows easy image Annotation & Segmentation and instant sharing of such Annotation & Segmentations. Using this Annotation & Segmentation 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 or-daring of objects in a scene, and increase the number of labels using minimal user supervision and images from the web.

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 [11], with 101 object classes (this was recently extended to 256 object classes [15]), the PASCAL collection [8], and the CBCL-street scenes database [5]. Many algorithms have been developed for image datasets where all training examples have the object of interest well-aligned with the other examples [39, 16, 42]. Algorithms that exploit context for object recognition [37, 17] 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.

TABLE OF CONTENTS

1. Introduction

Thousands of objects occupy the visual world in which we live. Biermann [4] estimates that humans can recognize about 30000 entry-level object categories. Recent work in computer vision has shown impressive results for the detection and recognition of a few different object categories [42, 16, 22]. 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 [10, 35] and multi-task learning [38]. 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 [39, 16, 42]. Algorithms that exploit context for object recognition [37, 17] 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 [2, 20] for relevant issues). Even algorithms requiring no supervision [31, 28, 10, 35, 34, 27] 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 [11], with 101 object classes (this was recently extended to 256 object classes [15]), the PASCAL collection [8], and the CBCL-street scenes database [5].

We wish to collect a large dataset of annotated images. To achieve this, we consider GUI-based data collection methods. GUI-based Annotation & Segmentation tools provide a way of building large annotated datasets by relying on the collaborative effort of a large population of users [43, 30, 29, 33]. Recently, such efforts have had much success. The Open Mind Initiative [33] 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 [30] is a commercial effort to provide an offline 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 offline game that is played by many users. The ESP game [43] pairs two random offline 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, Peekaboo [44] 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 IASTool, a database and an Annotation & Segmentation tool that allows the sharing of images and Annotation & Segmentations. The gui tool provides functionalities such as drawing polygons, querying images, and browsing the database. In the first part of the paper we describe the Annotation & Segmentation 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 & Segmentation process. In the third part we compare the IASTool dataset against other existing datasets commonly used for object detection and recognition.

2. IASTool

In this section we describe the details of the Annotation & Segmentation tool and the results of the offline collection effort.

2.1 Goals of the IASTool project

There are a large number of publicly available databases of visual, we do not have space to review them all here. However, we give a brief summary of the main features that distinguishes the IASTool 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 Section 4).

- 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 IASTool 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 IASTool database is designed to allow collected labels to be instantly shared via the web and to grow over time.

2.2 The IASTool GUI-based Annotation & Segmentation tool

The goal of the Annotation & Segmentation 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 Python drawing tool, as shown in Figure 1. When the user enters the page, an image is dis-played. 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 IASTool 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 Annotation & Segmentations portable and easy to extend.

The Annotation & Segmentation tool design choices emphasize simplicity and ease of use. However, there are many concerns with this Annotation & Segmentation 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

Figure 1. A screenshot of the labeling tool in use. The user is shown an image along with possibly one or more existing Annotation & Segmentations, 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 & Segmentation. The user may annotate as many objects in the image as they wish.

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 Annotation & Segmentations 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 "per-son", "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 Section 3.1, we de-scribe how to cope with the text label variability via WordNet [13]. All of the above issues are revisited, addressed, and quantified in the remaining sections.

A GUI 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 offline tool (this communication can in fact allow one to only download desired parts of the dataset), image manipulations, and other dataset extensions (see Section 3).

The images and Annotation & Segmentations are organized offline 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 Annotation & Segmentations come from two different sources: the IASTool offline Annotation & Segmentation tool and Annotation & Segmentation tools developed by other research groups. We indicate the sources of the images and Annotation & Segmentations in the folder name and in the XML Annotation & Segmentation 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 IASTool

We summarize the content of the IASTool database as of 30 August,2020. The app consists of drawing polygons, with any number of side polygons annotated using the gui tool and the polygons annotated offline.

As outlined above, a IASTool 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. Offline labels entered 2888 different descriptions for the 44059 polygons (there are a total of 4210 different descriptions when considering the entire dataset). Figure 2(a) shows a sorted histogram of the number of instances of each object description for all 111490 poly-gons[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 section 3.1 we will address the issue of unifying the terminology to properly index the dataset according to real object categories.

Figure 2(b) shows a histogram of the number of annotated images as a function of the percentage of pixels labeled per image. The graph shows that 11571 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 2(c) 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 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 & Segmentation tool went offline. We show the number of new polygons and text descriptions entered as a function of time. For this analysis, we only consider the 44059 polygons entered using the GUI-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 44059 polygons provided offline. For each object category, we sort the polygons according to the

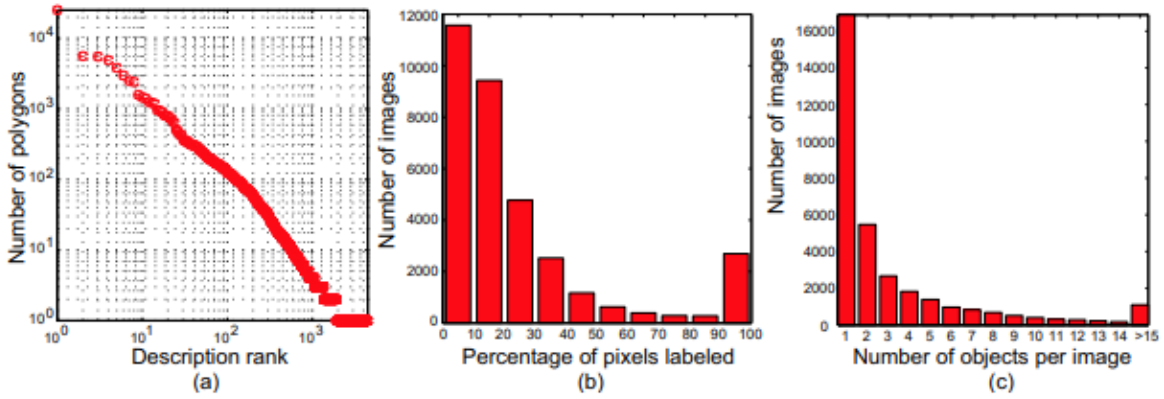


Figure 2. Summary of the database content. (a) Sorted histogram of the number of instances of each object description. Notice that there is a large degree of consensus with respect to the entered descriptions. (b) Histogram of the number of annotated images as a function of the area labeled. The first bin shows that 11571 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.

Figure 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.

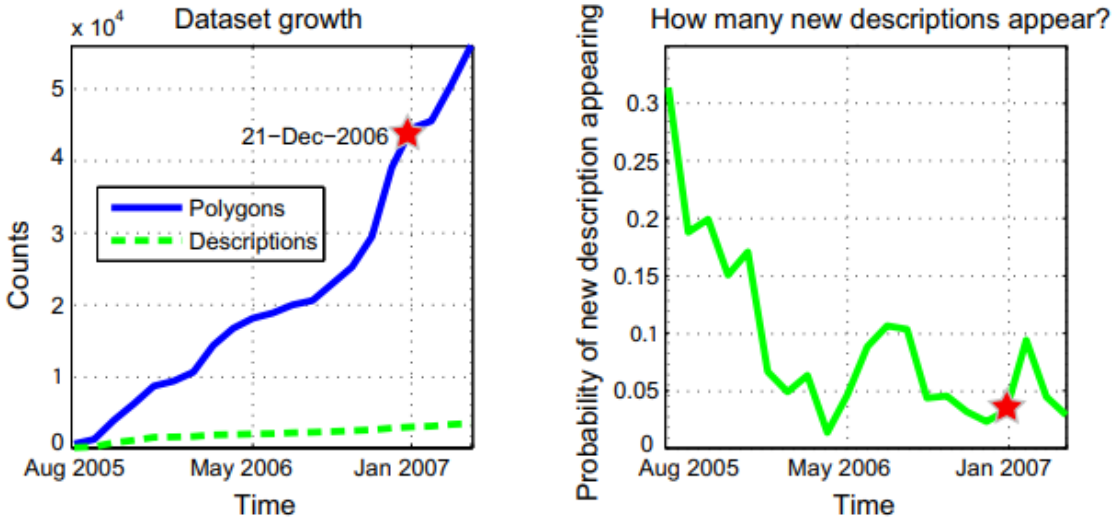


Figure 4. Evolution of the online annotation collection over time. Left: total number of polygons (blue, solid line) and descriptions (green, dashed line) in the LabelMe dataset as a function of time. Right: the probability of a new description being entered into the dataset as a function of time. Note 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.

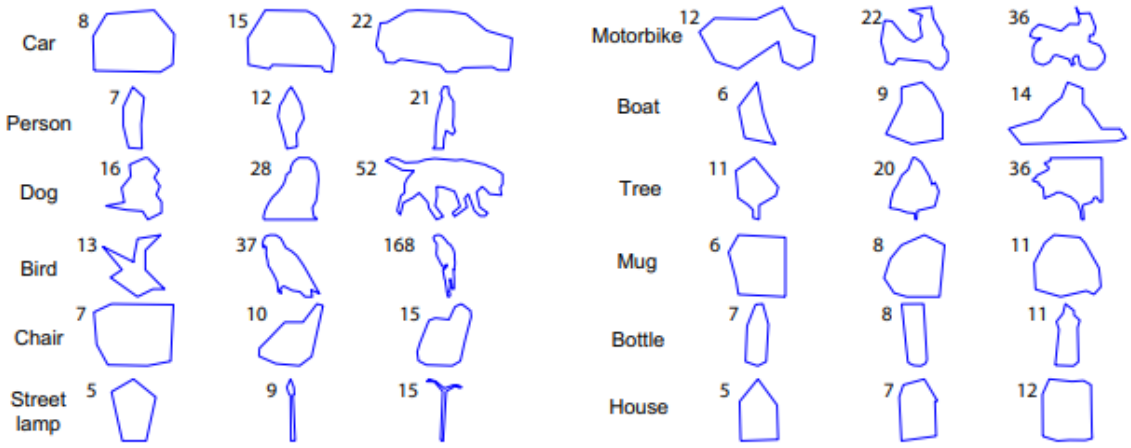


Figure 5. Illustration of the quality of the annotations in the dataset. For each object we show three polygons depicting annotations corresponding to the 25th, 50th, and 75th percentile of the number of control points recorded for the object category. Therefore, the middle polygon corresponds to the 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.

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.

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 IASTool 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

Figure 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.

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 resizing each image randomly.

3. Extending the dataset

We have shown that the IASTool 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 gui toolbox.

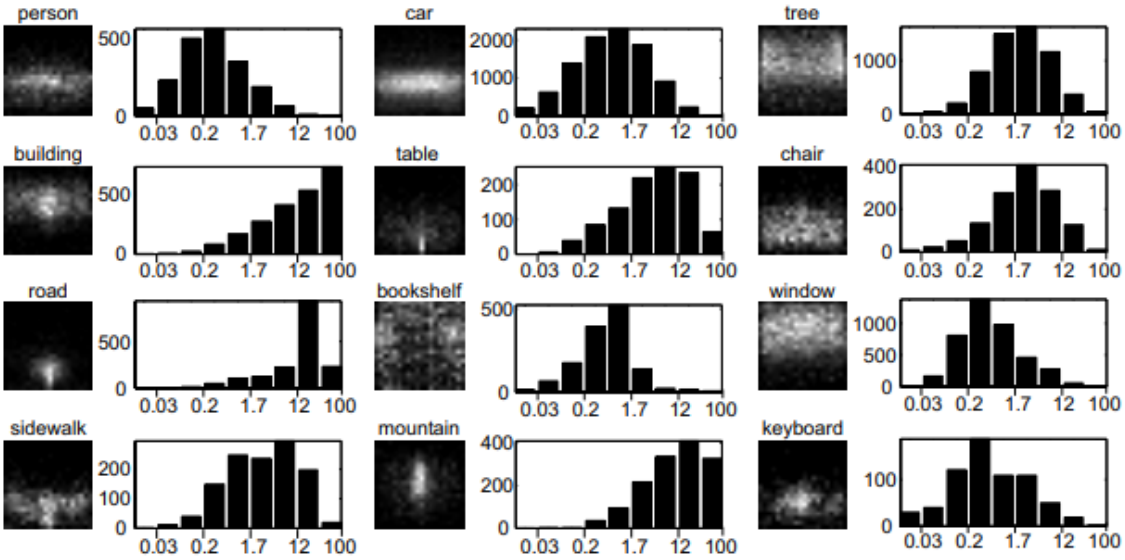


Figure 7. Distributions of object location and size for a number of object categories in the LabelMe dataset. The distribution of locations are shown as a 2D histogram of the object centroid location in the different images (coordinates are normalized with respect to the image size). The size histogram 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.

3.1 Enhancing object labels with WordNet

Since the Annotation & Segmentation tool does not restrict the text labels for describing an object or region, there can be a large variance 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 Annotation & Segmentations to incorporate superordinate categories, such as "animal", "vehicle", and "furniture".

We use WordNet [13], an electronic dictionary, to extend the IASTool 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 an IASTool 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 [18]. 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 IASTool 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. Figure 4), the manual sense disambiguation can be done in a few hours for the entire dataset.

person (27719 polygons)		car (10137 polygons)	
Label	Polygon count	Label	Polygon count
person walking	25330	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

Table 1. Examples of IASTool 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 represent the number of returned objects that have the corresponding description. Note that some of the descriptions do not contain the query words.

We extended the IASTool Annotation & Segmentations 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 choose 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%) unique IASTool descriptions found a WordNet mapping, which corresponds to 104740 out of the 111490 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 IASTool decreases over time (c.f. Figure 4).

We show the benefit of adding WordNet to IASTool to unify the descriptions provided by the different users. Table 1 shows examples of IASTool 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 Annotation & Segmentations with WordNet) are distributed across different IASTool descriptions. It is interesting to observe that all of the queries seem to

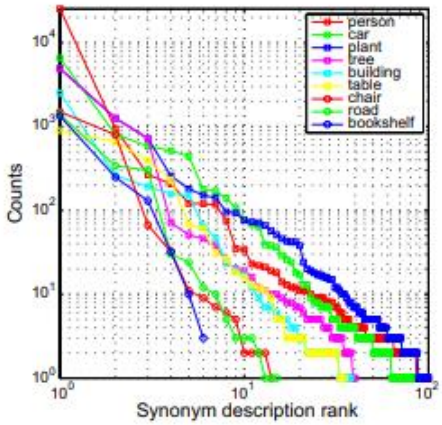


Figure 8. How the polygons returned by one query (in the WordNet-enhanced framework) are distributed across different descriptions. The distributions seem to follow a similar law: a linear decay in a log-log plot with the number of polygons for each different description on the vertical axis and the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

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 intointo IASTool 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. 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.

Category	Original description	WordNet description
person	27019	27719

car	10087	10137
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

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.

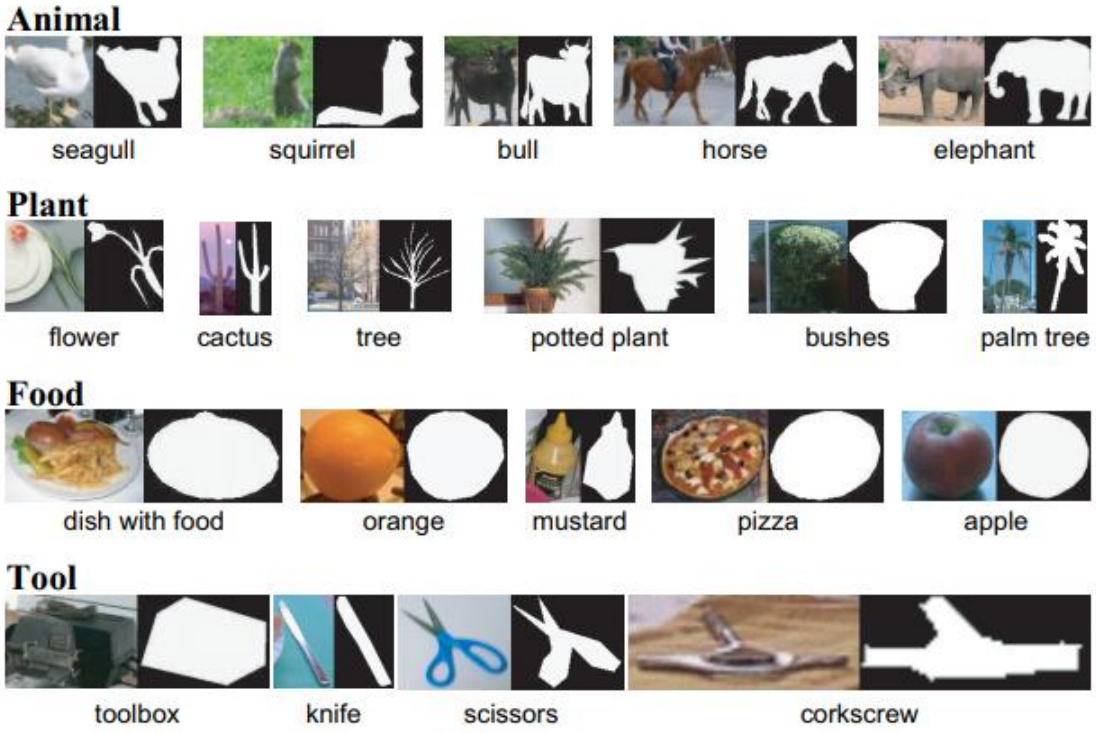


Figure 9. Queries for superordinate object categories after incorporating WordNet. Very few of these examples were labeled with a description that matches the superordinate category (the original LabelMe descriptions are shown below each image). Nonetheless, we are able to retrieve these examples.

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 Section 3.3.

We propose the following heuristic to discover semantically meaningful object-part relation-ships. 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 over-lap. 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

a 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 Section 3.3) and assign the part to the polygon with smallest depth that exceeds the intersection ratio threshold. Figure 11 gives some quantitative results on the number of parts per object and the probability with which a particular object-part is labeled.

Figure 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.

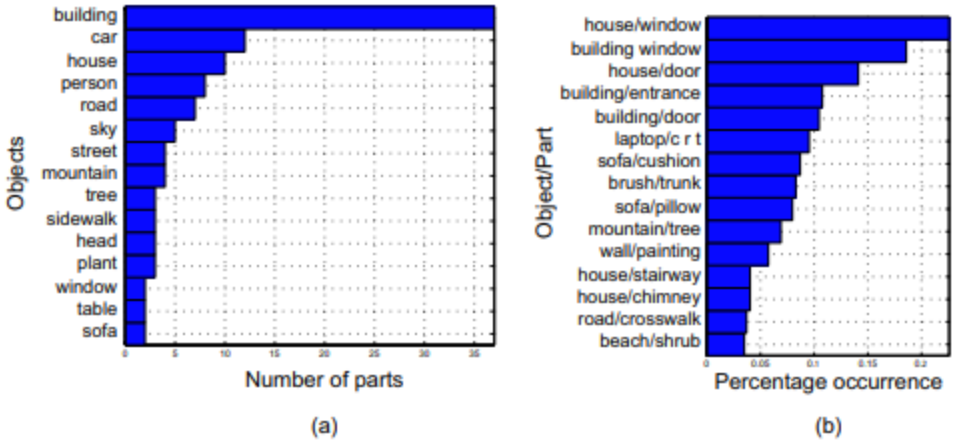


Figure 11. Quantitative results showing (a) how many parts an object has and (b) the likelihood that a particular part is labeled when an object is labeled. Note that there are 29 objects with at least one discovered part (only 15 are shown here). We are able to discover a number of objects having parts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

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:

- 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

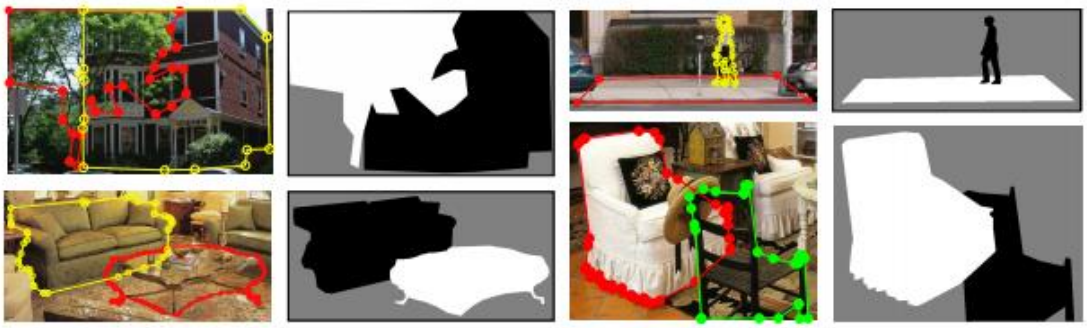


Figure 12. Each image pair shows an example of two overlapping polygons and the final depth-ordered segmentation masks. Here, white and black regions indicate near and far layers, respectively. A set of rules (see text) were used to automatically discover the depth ordering of the overlapping polygon 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.

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 [36] 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.

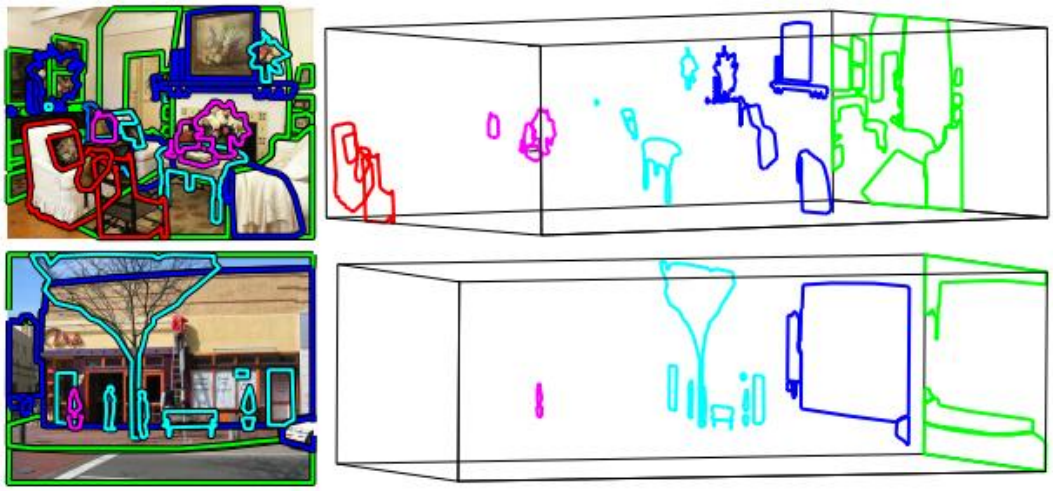
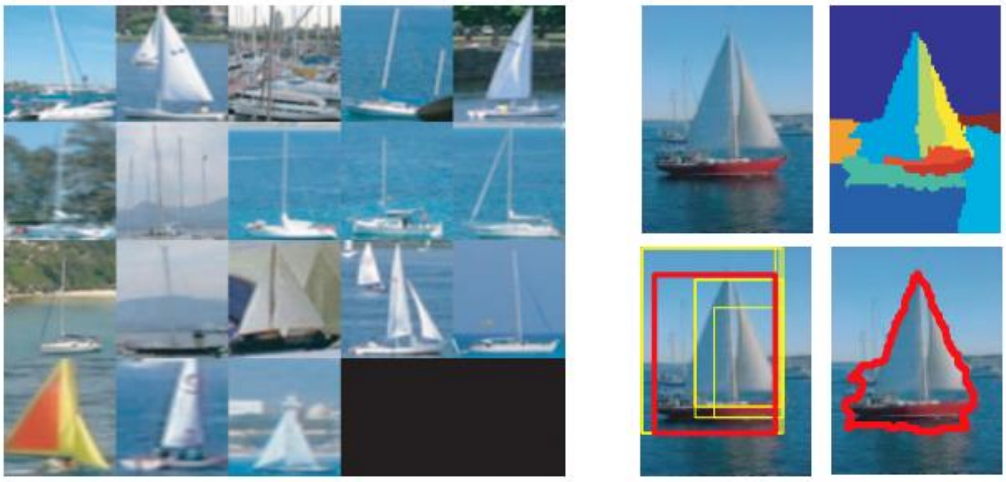


Figure 13. Decomposition of a scene into layers given the automatic depth ordering recovery of polygon pairs. Since we only resolve the ambiguity between overlapping polygon pairs, the resulting ordering may not correspond to the real depth ordering of all the objects in the scene.

3.4 Semi-automatic labeling

Once there are enough Annotation & Segmentations 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 [41]. A successful instance of this idea is the Seville project [1] where an incremental, boosting-based detector was trained. They started by training a coarse detector that was 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 IASTool to train a coarse detector to be used to label images obtained from offline image indexing tools. For instance, if we want more annotated samples of *sailboats*, we can query both IASTool (18 segmented examples of sailboats were returned) and offline image search engines (e.g. Google, Flickr, and AltaVista). The offline image search engines will return thousands of unlabeled images that are very likely to contain a sailboat as a prominent object. We can use IASTool 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



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

Figure 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 downloaded 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.

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

Figure 15. Enhancing web-basd 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.

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 [28], edge fragments [26], multiscale-oriented filters [24]). For the experiments presented here, we used the Gist features [24] (code available offline) to represent each region.

- Perform classification: train a support vector machine classifier [40] with a Gaussian kernel using the available IASTool 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 IASTool, we collected 18 sailboats, 41 dogs, 154 bottles, 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 offline queries [14, 3]. We used the detector trained with IASTool to sort the images returned by the offline query tools.

Figure 15 shows the results and compares the images sorted according to the ranking given by the output of the offline 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 im-ages downloaded from the web (for sailboats and dogs). The precision-recall graphs show that the score provided by the classifier provides a better measure of probability of presence of the queried object than the ranking in which the images are returned by the offline 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 [9], an ob-ject is considered correctly detected when the area of overlap between the ground truth bound-ing 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%

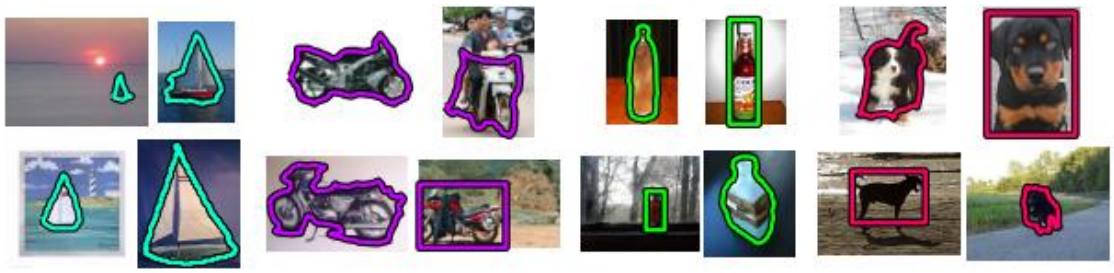


Figure 16. Examples of automatically generated segmentations and bounding boxes for sailboats, motorbikes, bottles, and dogs.

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 annota-tions. This is shown in Figure 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 detect. 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 [32]. 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 IASTool dataset against four annotated datasets currently used for object detection and recognition: Caltech-101 [12], MSRC [45], CBCL-Street Scenes [5], and PASCAL 2006 [9]. Table 3 summarizes these datasets. The Caltech-101 and CBCL-street scenes

Dataset	# categories	# images	# Annotation & Segmentations	Annotation & Segmentation type
IASTool	183	30369	111490	Polygons
Caltech-101 [12]	101	8765	8765	Polygons
MSRC [45]	23	591	1751	Region masks
CBCL-Streetscenes [5]	9	3547	27666	Polygons
Pascal 2006 [9]	10	5304	5455	Bounding boxes

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

provide location information for each object via polygonal boundaries. PASCAL2006 provides bounding boxes and MSRC provides segmentation masks.

For the following analysis with the IASTool 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 IASTool dataset, contributed Annotation & Segmentations and sequences.

Figure 17(a) shows, for each dataset, the number of object categories and, on average, how many objects appear in an image. Notice that currently the IASTool dataset contains more object categories than the existing datasets. Also, observe that the CBCL-Street scenes and IASTool datasets often have multiple Annotation & Segmentations per image, indicating that the images correspond to scenes and contain multiple objects. This is in contrast with the other datasets, which prominently feature a small number of objects per image.

Figure 17(b) 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 IASTool 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 IASTool 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 17(c) 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 IASTool 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 Annotation & Segmentations. Figure 17(d) shows the number of polygonal Annotation & Segmentations as a function of the number of control points. The IASTool dataset has a wide range of control points and the number of Annotation & Segmentations 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 Annotation & Segmentations consist of bounding boxes and region masks, respectively.

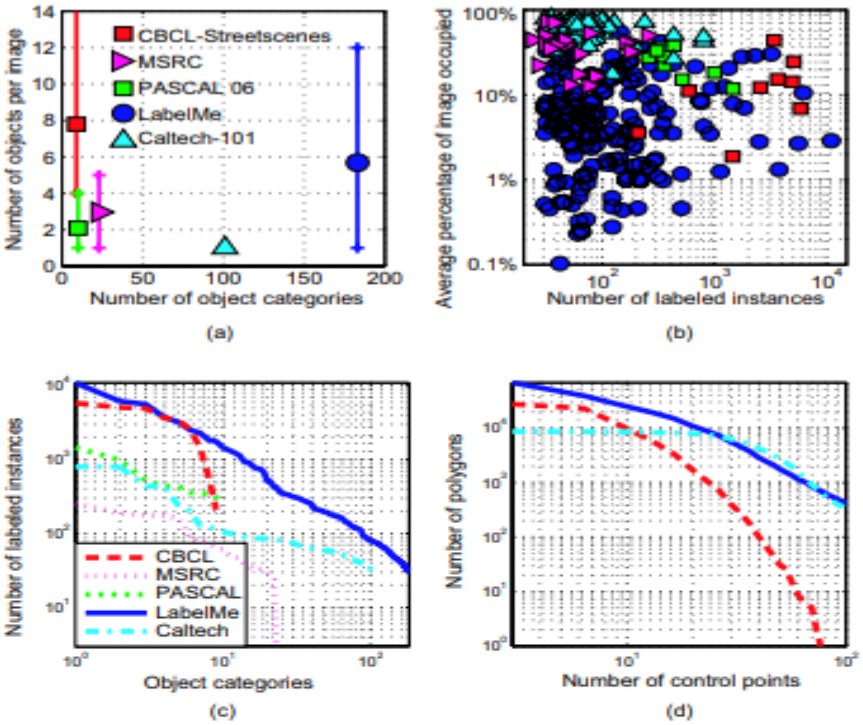


Figure 17. Comparison of five datasets used for object detection and recognition: Caltech-101 [10], MSRC [45], CBCL-Streetscenes [5], PASCAL2006 [9], and LabelMe. (a) Number of object categories versus number of annotated objects per image. (b) Scatter plot of number of object category instances versus 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).

5. Conclusion

We described a GUI-based image Annotation & Segmentation tool that was used to label the identity of objects and where they occur in images. We collected a large number of high quality Annotation & Segmentations, 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 offline 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 IASTool 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.

References

[1] Y. Abramson and Y. Freund. Semi-automatic visual learning (seville): a tutorial on active learning for visual object recognition. In *Intl. Conf. on Computer Vision and Pattern Recognition (CVPR05), San Diego*, 2005.

[2] Shivani Agarwal, Aatif Awan, and Dan Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(11):1475–1490, 2004.

[3] T. L. Berg and D. A. Forsyth. Animals on the web. In *CVPR*, volume 2, pages 1463–1470, 2006.

[4] I. Biederman. Recognition by components: a theory of human image interpretation. *Pyschological review*, 94:115–147, 1987.

[5] S. Bileschi. CBCL streetscenes. Technical report, MIT CBCL, 2006. The CBCL-Streetscenes dataset can be downloaded at `http://cbcl.mit.edu/software-datasets`.

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

[7] Owen Carmichael and Martial Hebert. Word: Wiry object recognition database. www.cs.cmu.edu/˜owenc/word.htm, January 2004. Carnegie Mellon University.

[8] M. Everingham, A. Zisserman, C. Williams, L. Van Gool, M. Allan, C. Bishop, O. Chapelle, N. Dalal, T. Deselaers, G. Dorko, S. Duffner, J. Eichhorn, J. Farquhar, M. Fritz, C. Garcia, T. Griffiths, F. Jurie, D. Keysers, M. Koskela, J. Laaksonen, D. Larl us, B. Leibe, H. Meng, H. Ney, B. Schiele, C. Schmid, E. Seemann, J. Shawe-Taylor, A. Storkey, S. Szedmak, B. Triggs, I. Ulu-soy, V. Viitaniemi, and J. Zhang. The 2005 pascal visual object classes challenge. In *First PASCAL Challenges Workshop*. Springer-Verlag, 2005.

[9] M. Everingham, A. Zisserman, C.K.I. Williams, and L. Van Gool. 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/`.

[10] L. Fei-Fei, R. Fergus, and P. Perona. A bayesian approach to unsupervised one-shot learning of object categories. In *IEEE Intl. Conf. on Computer Vision*, 2003.

[11] L. Fei-Fei, R. Fergus, and P. Perona. 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*, 2004.

[12] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Trans. Pattern Recognition and Machine Intelligence*, In press. The Caltech 101 dataset can be downloaded at `http://www.vision.caltech.edu/Image Datasets/Caltech101/Caltech101.html`.

[13] C. Fellbaum. *Wordnet: An Electronic Lexical Database*. Bradford Books, 1998.

[14] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning object categories from google's image search. In *Proceedings of the 10th International Conference on Computer Vision, Beijing,China*, volume 2, pages 1816–1823, October 2005.

[15] G. Griffin, A.D. Holub, and P. Perona. The Caltech-256. T echnical report, California Institute of Technology, 2006.

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

[17] D. Hoiem, A. Efros, and M. Hebert. Putting objects in perspective. In *CVPR*, 2006.

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

[19] Yann LeCun, Fu-Jie Huang, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of CVPR'04*. IEEE Press, 2004.

[20] B. Leibe. *Interleaved object categorization and segmentation*. PhD thesis, 2005.

[21] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *ECCV*, 2004.

[22] B. Leibe and B. Schiele. Analyzing appearance and contour based methods for object categorization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'03)*, Madison, WI, June 2003.

[23] Y. Li and L. G. Shapiro. Consistent line clusters for building recognition in cbir. In *Proceedings of the International Conference on Pattern Recognition*, 2002.

[24] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *Intl. J. Computer Vision*, 42(3):145–175, 2001.

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

[26] A. Opelt, A. Pinz, and A. Zisserman. A boundary-fragment-model for object detection. In *ECCV*, 2006.

[27] P. Quelhas, F. Monay, J. M. Odobez, D. Gatica-Perez, T. Tuytelaars, and L. Van Gool. Modeling scenes with local descriptors and latent aspects. In *IEEE Intl. Conf. on Computer Vision*, 2005.

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

[29] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. IASTool: a database and GUI-based tool for image Annotation & Segmentation. Technical Report AIM-2005-025, MIT AI Lab Memo, September, 2005.

[30] Flickr Photo Sharing Service. http://www.flickr.com.

[31] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering objects and their location in images. In *IEEE Intl. Conf. on Computer Vision*, 2005.

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

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

[34] E. Sudderth, A. Torralba, W. T. Freeman, and W. Willsky. Describing visual scenes using transformed dirichlet processes. In *Advances in Neural Info. Proc. Systems*, 2005.

[35] E. Sudderth, A. Torralba, W. T. Freeman, and W. Willsky. Learning hierarchical models of scenes, objects, and parts. In *IEEE Intl. Conf. on Computer Vision*, 2005.

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

[37] A. Torralba. Contextual priming for object detection. *Intl. J. Computer Vision*, 53(2):153–167, 2003.

[38] A. Torralba, K. Murphy, and W. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *CVPR*, 2004.

[39] M. Turk and A. Pentland. Eigenfaces for recognition. *J. of Cognitive Neuroscience*, 3(1):71–86, 1991.

[40] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1999.

[41] T. Vetter, M. Jones, and T. Poggio. A bootstrapping algorithm for learning linear models of object classes. In *CVPR*, 1997.

[42] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple classifiers. In *CVPR*, 2001.

[43] L. von Ahn and L. Dabbish. Labeling images with a computer game. In *Proc. SIGCHI conference on Human factors in computing systems*, 2004.

[44] L. von Ahn, R. Liu, and M. Blum. Peekaboom: A game for locating objects in images. In *In ACM CHI*, 2006.

[45] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictio-nary. In *IEEE Intl. Conf. on Computer Vision*, 2005. The MSRC dataset can be downloaded at `http://research.microsoft.com/vision/cambridge/recognition/default.htm`.