CS 131 Lecture 16: Recognizing Objects by Parts

Tyler Dammann, Patrick Mogan, Qiqi Ren, Cody Stocker, Evin Yang
Department of Computer Science
Stanford University
Stanford, CA 94305
{tdammann, pjmogan, qiqiren, cstock2, evin}@stanford.edu

1 Introduction

One overarching goal of computer vision is to be able to determine from an image of an object not just the object's class, but also relevant information about that object. Motivating examples include:

- From an image of a piece of sushi, can we determine how many calories it has?
- If we encounter a mountain lion in the wild, can computer vision software tell us the safest course of action?
- Given a chair, can we find out where online we could buy it?
- From an image of a mushroom, can computer vision software tell us if it's safe to eat?

In general, computer vision should be able to give helpful information about an object beyond just basic characteristics.

2 What can computers recognize today?

Computer vision algorithms can easily detect and recognize faces in images.

Object identification software like Google Goggles can find very specific kinds of objects (e.g. logos, landmarks, books, text, contact info, artwork), but can only find exact matches. Finding a generic object is much harder for today's systems.

3 What's next to work on?

A principle milestone in computer vision yet to be reached is universal class recognition. That is, we would like an algorithm that could, for an image of any object, identify its class. Examples include:

- For an image of an orange mug, we would like coffee mugs to be the result, but image search found similar images by looking for something orange in the center – not mugs.
- Given an image of someone with a gas pump, we'd like to recognize that the gas pump is in the picture, but Google image search gives us images of people standing in same area of picture, maybe wearing the same color clothing.

The PASCAL VOC challenge[1] taught models how to classify 20 classes of objects. However, a particular model can only recognize a finite number of classes of objects. A model cannot recognize an arbitrary object, such as a coffee mug, if it is outside the model's set of training classes. We want to be able to recognize everything, but there are a lot of "things", and the agreed-upon number of "things" is increasing over time, so it is hard to decide which ones to focus on. In addition, there is not even an agreed-upon number of "things" in the universe – there are 60K+ product categories

on eBay, 80K+ English nouns on WordNet, 3.5M+ unique tags on Flickr, and 4.1M+ articles on Wikipedia.

4 Big Data from the Internet

Because of the sheer amount of data available, the Internet should be able to teach us a huge number of things. Internet traffic has been steadily increasing, and vast majority (86%) is visual data.

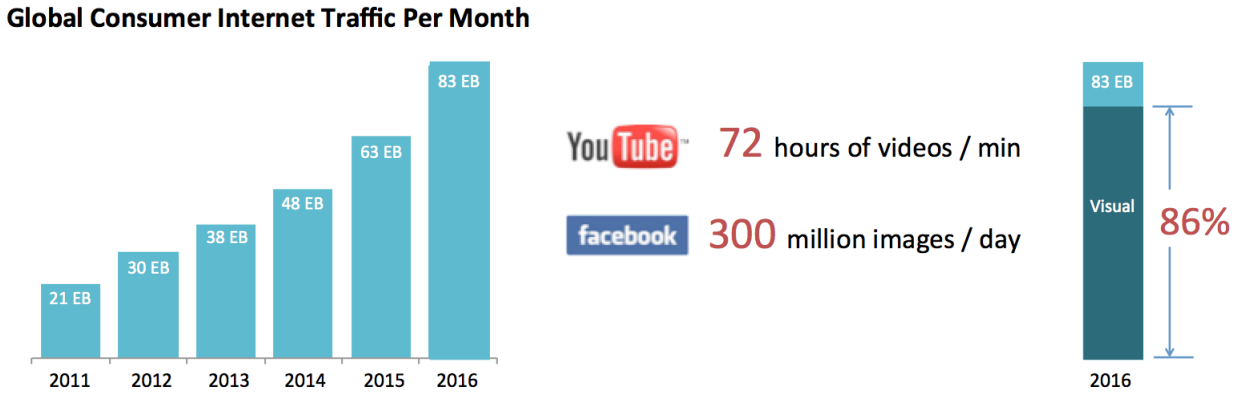

Figure 1: Global Internet Traffic. Source: Lecture 16

Visual data is known as the dark matter of the internet because we can't autoclassify it or auto-analyze it. Trying to categorize a photoshopped "pitbullfrog" shows that machines struggle to generalize, improvise, and extrapolate with the ease that humans do. Thus, images are not enough; just as important is leveraging the crowd, which can provide answers to questions that machines would otherwise have difficulty tackling themselves, such as in the following example:

Figure 2: Example Image Recognition Problem. Source: Lecture 16

Vitally, behind the big data is all of the users who are contributing their domain knowledge. When a human can't figure out the answer to a problem, it is easy to find other humans that can help figure it out. Image recognition systems, on the other hand, do not necessarily do this right now. The internet gives an environment in which these two resources can be combined.

5 ImageNet and Confusion Matrices

Since models are limited by the number of the classes in the training set, one possible method of advancing universal class recognition is to create datasets with many more than the 20 classes the PASCAL VOC provided.

ImageNet is a large image database created by Jia Deng in 2009 that has 22,000 categories and 14 million images. It has become so important to computer vision that most projects now train on ImageNet before tackling other challenges. Other well-known databases include: Caltech101 (9K images, 101 categories), LabelMe (30k images), SUN (131K images).

Deng applied four top classification methods from PASCAL VOC to ImageNet, but found that increasing the number of categories on which the models trained decreased their accuracy greatly. He plotted his findings as the following confusion matrix:

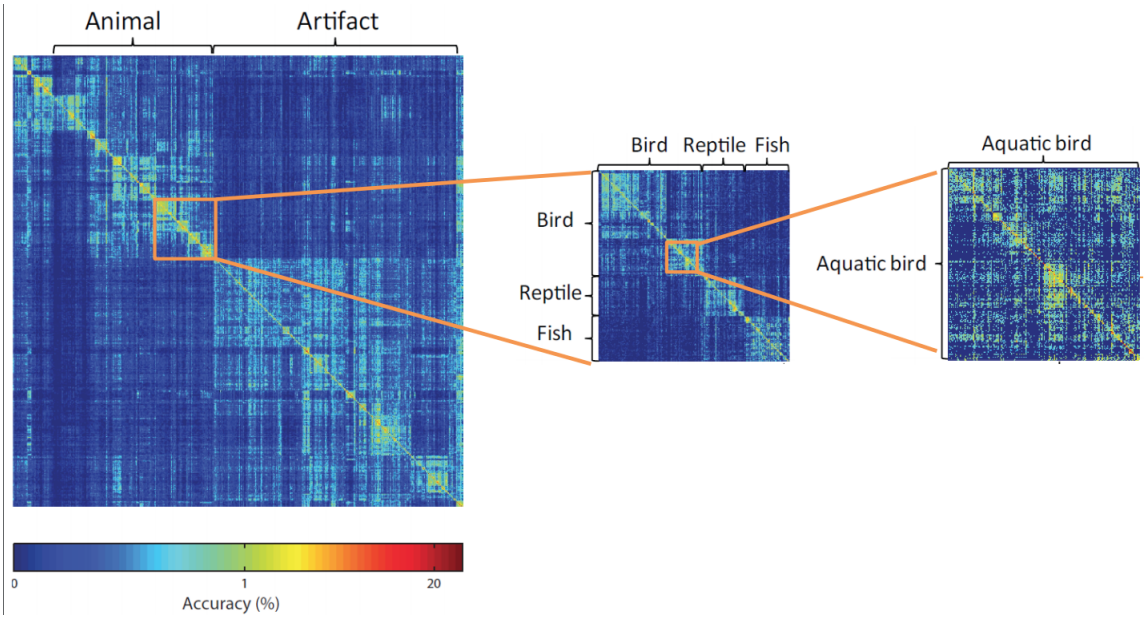

Figure 3: ImageNet Confusion Matrix. Source: Lecture 16

A confusion matrix plots categories on the x and y axis and measures how similar they are. If we had a perfect detector, the only bright areas would be along the center diagonal line since that would mean everything is only similar to itself.

We see from this matrix that models are more prone to making classification errors when presented with items whose categories are very similar; for example, models struggle with discerning between different types of aquatic birds, while it can more easily categorize birds versus dogs. In other words, distinguishing between "fine-grained" categories is very difficult, since the distance between those categories is much smaller than between larger categories.

6 Challenges and Solutions

6.1 Semantic Hierarchy

One method of attempting to solve the issue of correctly classifying similar classes without making wrong guesses is the idea of a "semantic hierarchy". An example can be seen in Figure 4. Essentially, a tree is created, where every child is a more specific subclass of the parent. As the category gets more specific, uncertainty increases. The system will attempt to be as specific as possible (as low down on the tree as possible) without an unacceptable probability of misclassifying

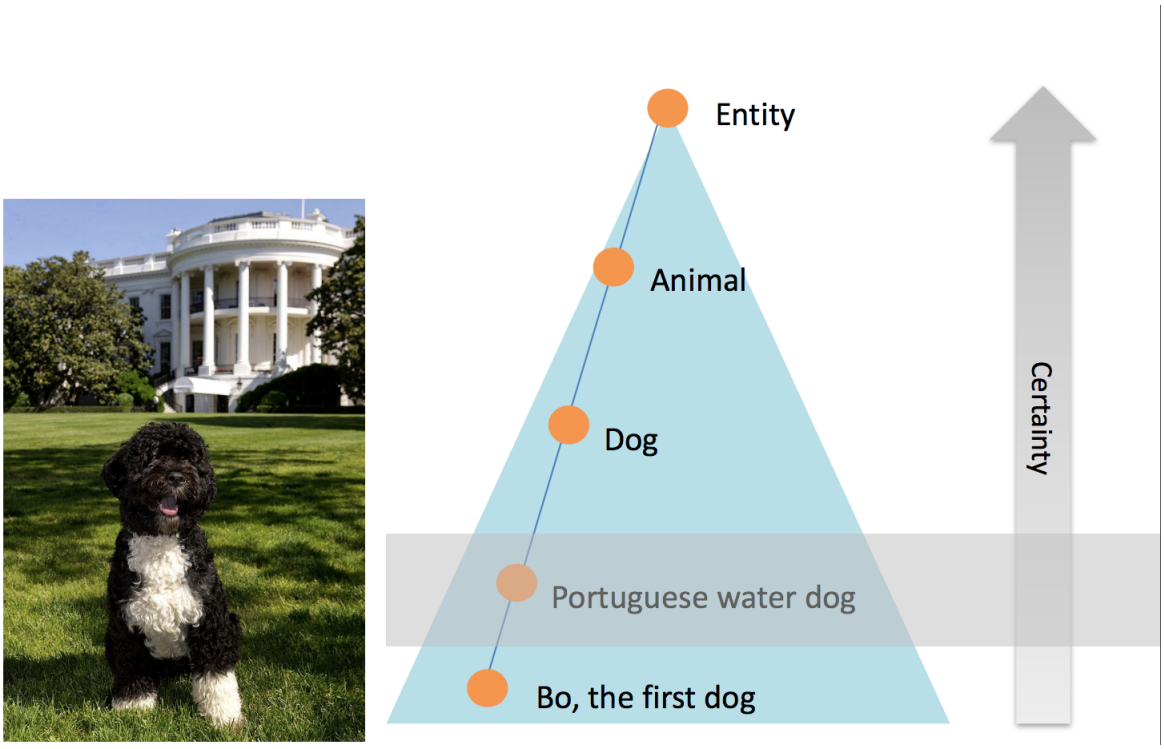

Figure 4: Example Semantic Hierarchy. Source: Lecture 16

the object. This concept is called "hedging" - the system tries to figure out where in the uncertainty tree to make a guess such that it is as informative as possible with few mistakes.

To define this problem formally, we will assume that the training and test set have the same distribution. In addition, we will assume that we have access to a base classifier g that gives a posterior probability on the hierarchy. We then define a reward function $R(f)$. We make $R(f)$ to have higher values at points farther down the hierarchical tree (when it classifies objects more specifically). Then, we also define an expected accuracy function $A(f)$. This function decreases as we move down the tree (since we are less sure about more specific guesses). Our problem statement is then to maximize $R(f)$ subject to the constraint that $A(f) \geq 1 - \epsilon$, where ϵ is a predetermined constant that represents the allowable error of our classifier over all of our examples.

However, potential non-optimal solutions (in terms of $R(f)$) can arise given this method. In order to fix this, we can define a global, fixed, scalar parameter $\lambda \geq 0$. For each node, we add λ to the reward value of that node, then normalize the posterior distribution. The process is then as follows:

1. Pick a λ.
2. Find the decision rule f with λ.
3. Measure the performance on the validation set.
4. Check if $A \approx 1 - \epsilon$. Repeat from step one if necessary.

We can use binary search to find the best λ quickly.

6.2 Fine-grained Classes

Existing work selects features from all possible locations in an image, but can fail to find the right feature. For example, the difference between the Cardigan Welsh Corgi and the Pembroke Welsh

Corgi is the tail. The computer may be unable to discover that this is the distinguishing feature. The easiest way to solve this type of problem is through crowd-sourcing.

What seems like a simple solution, however, is actually somewhat difficult – what is the best method for asking a crowd which features distinguish classes of images?

Bubble study: In order to detect the difference between smiling and neutral faces, we display only small bubbles of an image to people in an experiment to determine which bubbles allow them to detect when the image is of a person smiling. An example of how the bubble method works can be seen in figure 5. However, this method is very costly, in both time and money.

Another idea is to ask people to annotate images themselves (to simply point out where the im-

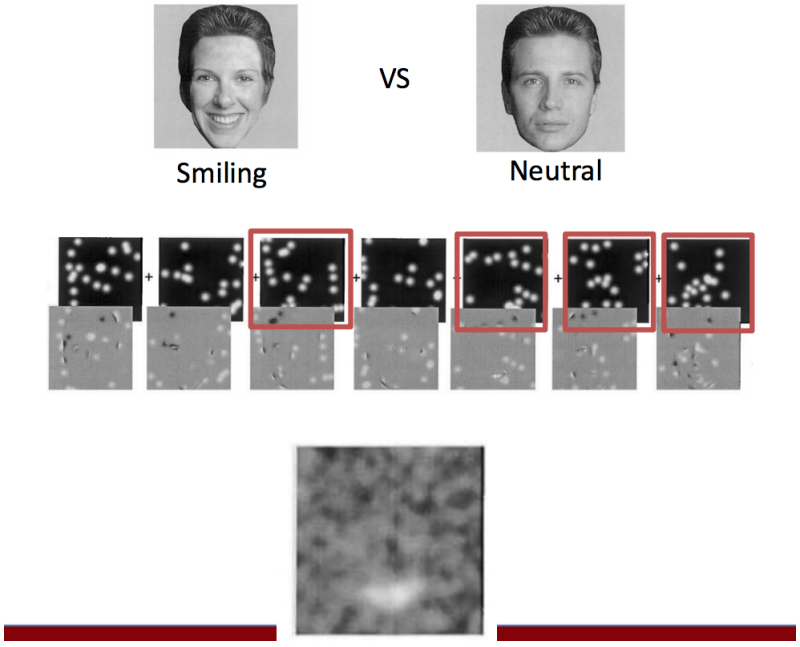

Figure 5: Bubble Method. Source: Lecture 16

portant features are). However, this method has no way of easily assuring the quality of these annotations.

Crowd-sourced bubble games: Online bubble games are similar in nature to the bubble study, but are fun and have monetary rewards for players. Notice that this solves both of the problems of the above methods – it is cost effective to have people play, while quality is assured by the reward system of the game. Two examples of games include the following:

- Peekaboom: Peekaboom was a two-player game in which one player reveals parts of an image, while the other player attempts to guess what the image represents. This game worked well for finding important parts of an image, but was not good for distinguishing between fine-grained classes (since often the important parts of the classes are the same).
- The Bubbles Game: This game, created by Jia Deng, is an online game in which players are given example images of two classes. Then, a blurred-out image is shown, and the player attempts to guess which class the blurred image belongs to, while revealing only small parts of the image. The less of the image revealed, the more reward is given to the player. This game is much better for finding differences between fine-grained classes, and it led to bubble heatmaps that pinpointed the features that distinguished fine-grained classes, as seen in figure 6. The creators of the game pooled bubbles from multiple images in a category together and the resulting information was provided to a model, which performed with higher precision than other models at the time.

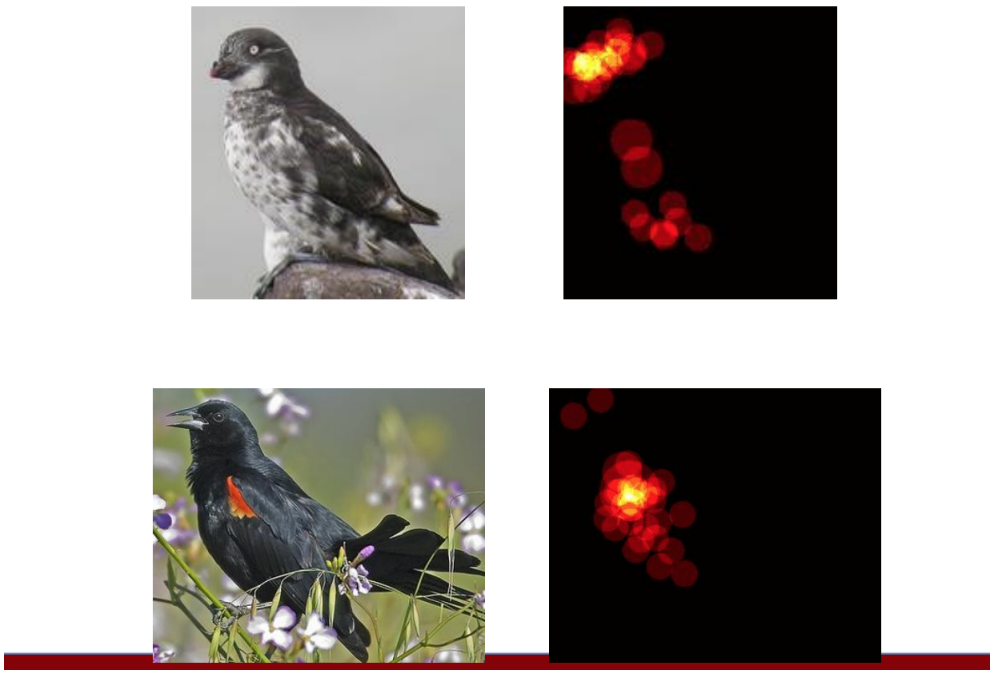

Figure 6: Bubble Heatmaps. Source: Lecture 16

References:
[1] Everingham, M. and Van-Gool, L. and Williams, C. K. I. and Winn, J. and Zisserman, A. *The PASCAL Visual Object Classes Challenge 2012. (VOC2012) Results*. Retrieved from: http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.