Tropel: Crowdsourcing Detectors with Minimal Training

Genevieve Patterson [1] Grant Van Horn[2] Serge Belongie[3] Pietro Perona[2] James Hays[1]
[1] Brown University, [2]California Institute of Technology, [3]Cornell University and Cornell Tech

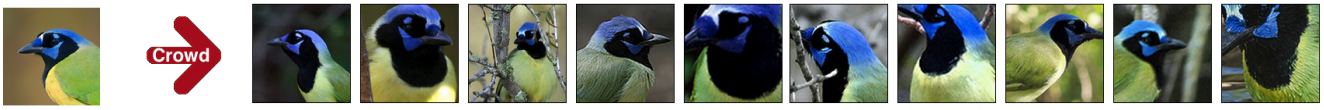

Figure 1: Top detections from a detector created with one seed example and trained by the crowd using our proposed method.

Abstract

This paper introduces the Tropel system which enables non-technical users to create arbitrary visual detectors without first annotating a training set. Our primary contribution is a crowd active learning pipeline that is seeded with only a single positive example and an unlabeled set of training images. We examine the crowd's ability to train visual detectors given severely limited training themselves. This paper presents a series of experiments that reveal the relationship between worker training, worker consensus and the average precision of detectors trained by crowd-in-the-loop active learning. In order to verify the efficacy of our system, we train detectors for bird species that work nearly as well as those trained on the exhaustively labeled CUB 200 dataset at significantly lower cost and with little effort from the end user. To further illustrate the usefulness of our pipeline, we demonstrate qualitative results on unlabeled datasets containing fashion images and street-level photographs of Paris.

Introduction

We live in a heavy-tailed visual world. There are visual concepts we encounter frequently (e.g. 'car') but a long tail of items which occur rarely (e.g. 'Caspian Tern', 'Louis Vuitton handbag', 'abandoned steel mill', 'frosted graham cracker cookies', 'Star-lord Halloween costume', 'Polynesian wedding ceremony' etc.). For almost any object type (e.g. 'guitar') there are hundreds of fine-grained subtypes that are of interest to some people (e.g. '1943 Gibson J-45', '1966 Fender telecaster').

An ambition of the computer vision community is to be able to detect *anything* in images, but how do users get the necessary training data? There exist impressive efforts to exhaustively annotate everything, e.g. ImageNet (Deng et al. 2009) in the case of objects, or to learn from the huge amount of weak supervision available on the internet, e.g. NEIL (Chen, Shrivastava, and Gupta 2013) and LEVAN (Divvala, Farhadi, and Guestrin 2014), or to empower domain experts to curate fine-grained databases, e.g. Visipedia (Perona 2010).

An alternative (which we adopt) is to admit that we can't anticipate the desires of all possible end users and instead only collect annotations as needed for a particular classifier. In this case *active learning* is a natural strategy to bypass the exhaustive annotation of a training dataset and instead iteratively label only those instances which are predicted to be most informative to the final classifier. However, this can still be tedious for moderately-sized datasets as a user repeatedly waits for the training and evaluation of each intermediate classifier before offering tens or hundreds of additional annotations.

In this paper we examine a scenario where an end user provides what is likely the minimum amount of supervision for non-trivial tasks – a single training example (e.g. one Puffin head shot to train a Puffin head detector). While we have seen rapid increases in recognition performance in recent years, a single training example is still grossly inadequate to train a reliable detector. On the other hand, the human visual system is surprisingly good at learning from a single example. Our proposed system exploits this gap. The single example provided by the end user is used to train *the crowd* who in turn provide hundreds of training examples to train a detector in an active learning setting. For a typical detector, a dozen crowd workers will collaborate intermittently over several hours as a computer vision system mines informative training examples and posts HITs to Amazon Mechanical Turk to ask for training labels.

In everyday life, people can learn to recognize visual concepts from a verbal description or a single visual example. To the authors' knowledge, exploiting this common human ability in the crowd context has not been addressed in the literature. In this paper we characterize the crowd's ability to train classifiers for rare visual phenomena given minimal training. We create over 200 detectors using the Tropel pipeline in several different visual domains.

With Tropel, we deliver a system that can create classifiers on demand, without a previously labeled dataset. This lowers the bar to entry for casual end users. Without specialized knowledge of how to design detectors for a particular visual domain (clothing, animals, architecture, etc.), users can employ Tropel to create detectors with minimum startup requirements. In this paper we seek to push the limits of what the minimum initialization can be for detector creation. We use the Tropel system to investigate how crowd active learning enables users to create useful detectors with the minimum effort or expertise in either computer vision or the vi-

Copyright © 2015, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

sual domain of interest.

In order to accomplish the goals of the Tropel project, the problems before us are:

1. What is the fewest number of user-submitted training examples required to show the crowd what they should annotate?
2. How specialized a concept can the crowd be trained to understand given the high turn over of workers for this type of task?
3. How can the responses of workers be cheaply evaluated and combined?

Related Work

To the authors' knowledge, this paper explores question 1 more directly and extensively than previous literature. Experiments with Tropel also explore questions 2 and 3 in a novel context.

Obtaining high quality image labels from crowdsourced annotators, who are often untrained and of unknown expertise, is an open research topic (Welinder et al. 2010; Gomes et al. 2011; Little et al. 2009). Active learning has been used to label objects that are *easy for non-experts recognize* such as pedestrians and PASCAL Visual Object Classes (Abramson and Freund 2004; Settles 2010; Collins et al. 2008; Vijayanarasimhan and Grauman 2011). Collins et al. (Collins et al. 2008) and Vijayanarasimhan and Grauman(Vijayanarasimhan and Grauman 2011) both use active learning with crowdsourced respondents as part of larger pipelines to create labeled datasets and detect the PASCAL challenge object categories.

However, both of these active learning systems need upwards of a hundred labeled training examples per category at initialization. The systems described in (Abramson and Freund 2004; Welinder et al. 2010; Collins et al. 2008; Vijayanarasimhan and Grauman 2011) also require laborious attention to the crowd workforce. The workers themselves are modeled, measured, solicited or cast out based on analytics collected by researchers. The goal of (Vijayanarasimhan and Grauman 2011) was to create detectors for the common objects of the PASCAL dataset ('bike', 'sheep', 'bottle', etc.). Crowd workers are able to identify this type of everyday item with little or no training. We explore the broader problem of how to use a crowd to train detectors capable of successfully identifying *anything* with the minimum amount of crowd training and quality control.

Crowd workers have been incorporated in the classifier creation process before. Deng et al. demonstrated a method for using the crowd to learn discriminative mid-level features for performing K-way classification on the CUB 200 dataset (Deng, Krause, and Fei-Fei 2013). Their approach uses a gamified interface to ask workers to identify the distinguishing elements between two easily confused categories. Deng et al. showed how non-technical crowd workers could be used to make fine-grained distinctions. While we also demonstrate results on the CUB 200 dataset, we do not use the ground truth categorical labels at training time and we are not necessarily interested in K-way image categorization – we aim simply for one detector from one training example in any visual domain.

The Tropel pipeline employs a large number of workers to answer small, simple-for-humans questions. Crowd users simply click on images that contain the query object. We require less of the worker than other successful active learning methods that ask uses to segment the relevant object or draw a bounding box, such as Vijayanarasimhan and Grauman (Vijayanarasimhan and Grauman 2011). However, we are asking the worker to concentrate on more specialized objects than the PASCAL VOC categories, as in (Vijayanarasimhan and Grauman 2011).

Research into crowd micro-tasks has recently been described by (Little et al. 2010; Chilton et al. 2013; Dow et al. 2012). Both Little et al. and Dow et al. show that minimal training helps crowd workers complete tasks more successfully than they would have been able to without training (Little et al. 2010; Dow et al. 2012).

In a rigorous investigation, Welinder et al. propose a method for accurately predicting the expertise and bias of a crowd worker (Welinder et al. 2010) for visual tasks. Unfortunately, the method introduced in (Welinder et al. 2010) assumes that a labeled validation set is available and it is possible to interact with an individual crowd worker over a number of questions. In the scenario addressed in this paper, the training dataset is assumed to be unlabeled and we have no expectation of retaining a worker for long enough to confidently establish their expertise. We investigate several approximations of the Welinder et al. method that improve detector accuracy over simple majority consensus among the workers. We concentrate on examining how to exploit the human ability to learn from a small number of examples to alleviate the lack of expertise in crowd workers.

We believe that Tropel is the first system to exploit the *combined efforts* of an end-user and the crowd via active learning. In the next section, we describe the details of the Tropel system and demonstrate its potential for easy parallelization and cost effective creation of large numbers of detectors.

Bootstrapping Classifiers

Our pipeline addresses a typical problem for end users. An end user, Jon, has access to a large set of images. Jon wants to search these images for a specific visual event, and begins his search from a positive example image. For example, he has a catalog picture of a jacket that he would like to buy. He wants to search a dataset of fashion images to find outfits where other people are wearing this type of jacket.

Tropel bootstraps detector training for user-specified objects of interest. The full input to Tropel is a single positive training example, an appropriate unlabeled database to learn from, and optionally a text label for the concept (e.g. 'Puffin head' or 'stiletto heel'). Later experiments in this section vary the number of initialization images. Equipped with a set of test objects and corresponding example images, the operation of the Tropel pipeline is formalized in Algorithm 1.

To begin the training process, each classifier is seeded with one user-submitted cropped image of the given object

Input: Dataset $\mathcal{D}$ of unlabeled images, items to detect A
Output: Classifiers C for items A

$A \Leftarrow$ item ▷ acquired through consultation with end user
$S_i \Leftarrow$ seed exemple of A
$\mathcal{NN}$ = `NearestNeighbors`$(S_i, \mathcal{D})$ ▷ 200 nearest neighbors of the seed example in $\mathcal{D}$
N_0, P_0 = `crowdQuery`$(\mathcal{NN})$ ▷ initial active query, crowd selects N_j negative and P_j positive images from the set of the seed example's nearest neighbors
C_0 = `svmTrain(` $S_i \cup P_0$`,` N_0`)`
$\mathcal{D}_j = \mathcal{D} \setminus \{x : x \in P_0 \cup N_0\}$
$j = 1$ ▷ current iteration counter
repeat
  Q_j= `sortDetections`$(C_j, \mathcal{D}_j)$ ▷ get top 200 detections by C_j in $\mathcal{D}_j$
  $N_j, P_j = N_{(j-1)}, P_{(j-1)} \cup$ `crowdQuery`(Q_j) ▷ crowdsourced active query
  C_j = `svmTrain(` $S_i \cup P_j$`,` N_j`)`
  $\mathcal{D}_j = \mathcal{D}_j \setminus \{x : x \in P_j \cup N_j\}$
  $j+ = 1$
until `convergence()`
return C, A

Algorithm 1: Crowd Active Learning with Minimal Initial Labeled Training Examples

or part. Initially, our system only possesses the seed example and a set of millions of training patches that have no associated class labels. In the first stage of crowd training, we ask the crowd to annotate the first 200 nearest neighbor patches of the seed example (we do not train an initial classifier because we have no trustworthy negative examples). These 200 nearest neighbor image patches are the closest patches to the seed examples out of the millions of image patches extracted from all images in the training set. The nearest neighbors are found using L2 distance in the 200-dimensional DeCAF-derived (Donahue et al. 2013) feature space used throughout the system. We make the assumption that the nearest neighbor patches are more likely to contain true positives or negative examples that closely resemble true positives than randomly sampled patches.

The response to this first active query provides the positive and negative training examples needed for the first iteration of the classifier. At each iteration of the active learning pipeline we train a linear SVM.

In the first iteration, where workers are reviewing nearest neighbor patches, and in later iterations, where workers are evaluating the top classifier detections, we only as workers to look at a maximum of 200 images patches. We would prefer to show more images at a time to collect more training examples, but obviously this can lead to worker fatigue if we ask them to look at too many images.

Once the first classifier is trained, the next round of active learning can begin. The first iteration classifier evaluates all of the image patches in the original training set minus any patches that have already been annotated. Until the detector converges to the ideal detector, the top 200 most confident detections are likely to contain the strongest misclassifications. Thus, we ask the crowd workers to evaluate the top 200 detections on the training set. As the iterative retraining approaches convergence, the top detections are more likely to be correct, but in the initial stages these top detections will provide hard negatives that rapidly shrink the version space. To easily compare detector improvement, we set a hard limit for the convergence criteria — 5 active learning iterations. The most confident detections for each iteration's classifier are presented to the crowd using the user interface shown in Fig. 2. The crowd we query for the following experiments is from Amazon Mechanical Turk (AMT).

Tropel is meant to be classifier agnostic. We use linear SVMs because they are lightweight to store and train. The choice of feature or image representation likely has a bigger impact than the final classifier.

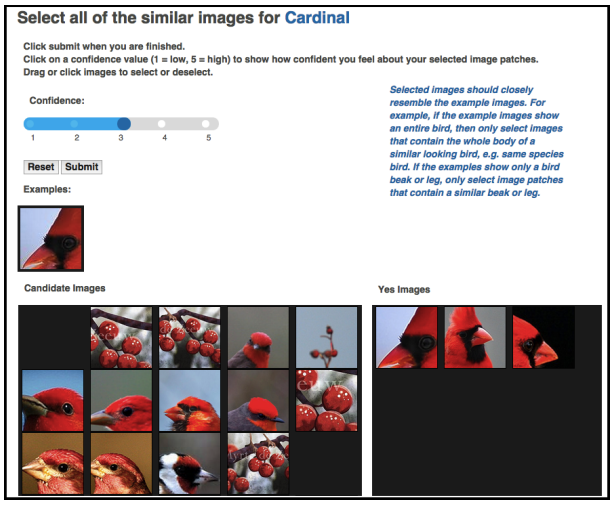

Figure 2: *Active Query User Interface*. The UI shown in this figure implements the function `crowdQuery()` in Alg. 1. Users click on the image patches they believe are visually and semantically similar to the example images. The column titled "Candidate Images" is scrollable so that users can view all 200 query images.

At each iteration of the active learning process, untrained crowd workers view a set of "Candidate Images". These are the top 200 detections from the current active learning classifier. The patches shown are pruned so that none of the patches have an overlap of greater than 0.3. The overlap ratio is defined by the area of the intersection divided by the union of the two patches. The workers select all of the patches that they believe match the seed example(s).

The interface in Fig. 2 also asks users to self-report their confidence in the accuracy of the selected images patches. The informativeness of this worker-supplied meta-data will be addressed in later sections.

Three workers independently answer each active learning query. If 2 out of 3 workers agree that a cropped image patch is a positive or negative case, that crop is added to the positive or negative training set for the next iteration's classifier. Crops that are not agreed upon are returned to the set of

queryable images.

In the Performance Comparison section we compare the average precision of detectors created by a crowd given a range of visual training examples. In the Worker Consensus Protocol Comparison section we examine the effect of weighting the votes from different workers by several metrics. We also compare the accuracy of individual workers to their time per hit, number of hits completed, and self-reported confidence.

Experimental Evaluation - CUB dataset

For the purposes of investigating our pipeline, we used the head part annotations that accompany the CUB 200 dataset (Wah et al. 2011). All of the images in this dataset have a head location annotated by a crowd worker. The larger part of these annotations were verified by experts. We obtained our set of example seeds by cropping 75×75 pixel head patches centered on those locations. For the fashion item tests a member of our team manually cropped 10 example patches for 10 different items of clothing and accessories.

We use a coarse-to-fine sliding window classifier. Tropel poses queries over the set of all sliding windows in the training set. The minimum window size is 75×75 pixels. Image patches are cropped at 4 different scales, up to 200×200 pixels. Typical image resolution is 500 pixels per side. The set of sliding window patches constitutes the training set for our active learning system.

We represent each patch with CNN features calculated using the pretrained DeCAF network from (Donahue et al. 2013). This network was trained on ImageNet which contains images of both birds and clothes. None of the ImageNet images are in our experiment datasets, which prevents pollution of the test images with images that were used to train the CNN. We reduce the dimensionality of the DeCAF feature by using only the first 200 activations from the last fully connected layer as suggested by Razavian et al. (Razavian et al. 2014) when using a pretrained network on a novel dataset. This dimensionality subsampling is especially helpful for a detection task where features are computed and stored for millions of sliding windows.

We first evaluate the ability of our system to build detectors for the heads of 200 different bird species. We compare the performance of our crowd active learning pipeline with a baseline detector.

While our pipeline is designed to train detectors from unlabelled datasets, we use the CUB 200 (Wah et al. 2011) dataset to quantify the accuracy of the learned detectors. We train classifiers for the heads of all 200 types of birds in the CUB 200 dataset, which contains 11,788 images split nearly evenly between training and test sets. The CUB 200 database contains ground truth part locations for the head of each bird. The seed examples for each bird head classifier were randomly selected from these labeled parts. Figure 3 shows the output of several bird head detectors at different stages of the Tropel pipeline, illustrating improving detector performance as the number of training iterations increase.

The box plot in Fig. 4 shows the iterative improvement of all 200 bird head detectors. Iterations 2-5 are shown as

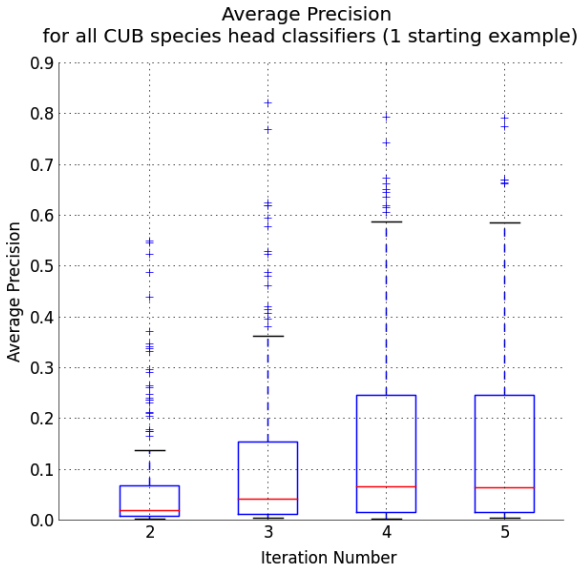

Figure 4: *Detector Average Precision at different iterations of the pipeline*. Each species has approximately 28 true positives in a dataset of millions of image patches. This plot gives an impression of the improving precision and recall with increasing iterations. Average Precision, the average value of the precision over the range of recall from recall = 0.0 to recall = 1.0 for each detector, is used for brevity.

no classifier is trained during iteration 1. The precision of each detector is calculated by counting true positives in the 30 most confident detections in the CUB test set. Note that there are at most 30 positive bird head bounding boxes for any species in the test set. Detections that overlapped by a ratio of greater than 0.3 were removed from the set of most confident detections to eliminate multiple detections of the same bird.

Some classifiers drastically increase in accuracy, most improve more modestly, and the bottom 25% of detectors hardly improve. The detectors that fail to improve are often hindered because those birds bear striking visual similarity to related species in the species taxonomy, e.g. many type of Sparrows look strongly similar. This drift to identifying related birds, and thus similar visual phenomena, is discussed in the Hierarchical Similarity section.

Performance Comparison

First, we compare Tropel to a typical computer vision approach – detectors trained only with trustworthy annotations and no active learning. The baseline classifier is a linear SVM trained with all the positive head patch examples in the training set and 10,000 randomly selected negatives. While we refer to these as 'baseline' classifiers, they could be expected to perform better than our pipeline because they train on crowd annotations that have been cleaned up and verified by experts. All detectors are evaluated by their detection average precision score on the CUB test set. We calculate detection AP using the PASCAL VOC standard, with the alter-

Figure 3: *Example detections at different iterations of the Tropel pipeline*. This figure shows the top 5 detections of 4 bird head classifiers. The species from left to right are: *Green Violetear, Vermillion Flycatcher, Sooty Albatross,* and *Gray-crowned Rosy Finch*. The first row in each block of images shows the seed examples used to start the pipeline. The following rows are the top 5 most confident detections on the training set. The different rows show the top 5 test detections from the first, second and fifth rounds of active learning.

ation that detections may be counted as true postives if they have an overlap ratio of 0.3 instead of the PASCAL threshold of 0.5 (Everingham et al. 2010).

The detection AP scores for the Tropel detectors are plotted against the performance of the baseline detectors in Fig. 5. We also show the performance of crowd classifiers when the workers are given slightly more training – 5 examples instead of 1. As expected, the AP scores for the detectors with 5 seed examples more closely approach the scores of the ground truth trained detectors. Overall, our crowd active learning process seeded with one or five training examples approaches the performance of the traditional detection pipeline. In Fig. 5 the average precision scores may appear low, however it is important to notice the difficulty of this baseline as the linear SVM trained on the fully annotated training set also has relatively low AP scores.

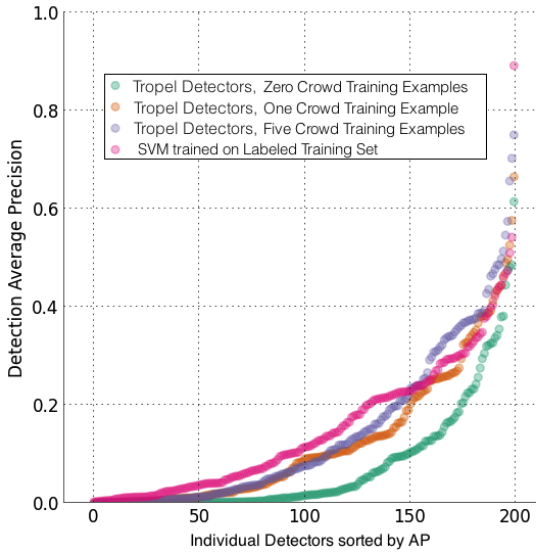

Figure 5: *Detector Average Precision*. This plot shows all of the 200 head detectors sorted by score for each method. Overall, this graph indicates that our methods approach the performance of ground truth, and more training examples for the crowd leads to higher AP.

The points plotted in Fig. 5 were obtained as follows: 1) a detector for each of the 200 bird species was trained using each of the 4 methods, 2) we calculated the average precision for all 800 detectors, 3) for each training pipeline, we took the 200 associated detectors and plotted their AP scores sorted from worst-to-best performance. For each of the trend lines, the ordering of the plotted detectors is different. We decided to plot each trend line in strictly ascending order to give the general impression for each training pipeline of how many classifiers performed badly, average, and well.

Our overall observation is that bird detectors that succeeded had one or more of these traits: 1) the bird had strong lines and sharp gradients in the head pattern, 2) the original example for the Turkers was an iconic-looking, well framed image, 3) the bird had no relatives that looked nearly identical.

Figure 5 shows that seeding the active learning process with five examples works slightly better than using a single example. There are two reasons for this – the candidate image patches shown to the workers in the first iteration are more likely to have positive examples and the human workers themselves are better able to pick out the positive examples because they have more training examples. That said, the difference between one and five examples ends up being fairly small (and critically they both perform nearly as well as detectors trained on exhaustive ground truth annotations). This raises the question of whether we could use even less supervision than a single example. Is it possible to train the crowd with no visual examples?

In a zero example training scenario there is no visual training at all. In our case that could mean a text label instead of a seed patch. Omitting the seed patch entirely is not viable in our current pipeline because we would have no reasonable initialization for the active learning. We could resort to showing random image patches, but for any given concept (e.g. 'cardinal head') there would likely be no positive examples for the crowd users to select and thus it is unlikely they could improve the detector even if they happen to know the concept.

Still, it is interesting to know what happens if we use the seed example to initialize the active learning but hide the seed example from the crowd workers. This gives us some

indication for how much 'training' the humans are getting from a single visual example. For this experiment, we initialize our pipeline identically to the one-example detectors – one initial seed patch was used to find nearest neighbor patches for the first active query. However, the crowd never sees the seed patch. The crowd workers are show a text description of what to look for, i.e. the species name. The UI for this zero crowd training experiment is the UI from Fig. 2 with the example image removed.

The zero example detectors perform strictly worse than the one example version with the same initialization, verifying that it was important to 'train' the crowd workers instead of expecting them to know the concept already (See Figure 5). Two notable outliers among the zero shot detectors are the detectors for the *Green Jay* and *Horned Puffin*. The initialization for these birds was unusually good (because they are distinctive birds) so it is likely that the crowd 'learned' what these birds looked like just by noticing what type of bird dominated the candidate patches. The Green Jay is a particular easy bird to guess – no other bird in the dataset has a similar distinctive neon green color, and so the workers who simply clicked on all green birds would have been making accurate selections.

It is important to note that in many low performing cases the appearance of the bird species often varies greatly for males, females, mating birds, etc. For example the *Black Tern* changes head color from black to white with a black eye-patch when the bird enters the mating stage. We could not train workers for this wide variety of appearance in the one example training setting. The examples for the 5 example detectors were selected at random from the ground truth head patches, and may not have captured all appearance variations.

As an additional experiment, we changed the user training examples to include negative examples. For each of the 200 bird species, we showed the crowd workers one positive example and 3 negative examples. The negative examples were selected from other species in the same family, so as to best show the subtle differences that distinguish one species from its cousin species. Overall this approach had little to no effect on the average AP score across categories. Table 1 shows 5 detectors where the negative examples helped workers and 5 where the negative examples were unsuccessful. In the cases where the negative examples didn't improve performance the workers may have been confused by the fact that the incorrect species birds were visually similar to the correct species. Generally these unsuccessful cases had very few new positive examples selected by workers.

Bird Species	Average Precision	
	With Negative Crowd Training	Only Positive Crowd Training
Groove billed Ani	0.12	0.00
Eastern Towhee	0.10	0.01
Least Auklet	0.17	0.06
Purple Finch	0.21	0.12
Painted Bunting	0.58	0.50
Red faced Cormorant	0.00	0.25
Rhinoceros Auklet	0.02	0.13
Sooty Albatross	0.03	0.25
Brewer Blackbird	0.00	0.16
Brown Creeper	0.00	0.26

Table 1: *Crowd Training with Negative Examples*. This table compares detector average precision (AP) two types of crowd worker training. In the left results column, the detectors were trained by workers who were shown both a positive example of the species and 3 incorrect examples from birds of cousin-species. In the right column, workers were only shown one positive example.

Detector Type	Avg. AP Score
Active Learning with Oracle	0.095
Tropel with One Example	0.106
SVM trained on Labeled Dataset	0.157
Active Learning with Random Responses	1.2e-4
Chance	3.9e-4

Table 2: *Comparison to Active Learning Baseline*. Averages are taken over 40 detectors for randomly selected bird species.

Comparison to Active Learning Baseline

As an alternative baseline, we also compare Tropel to a canonical active learning pipeline. Our "Active Learning with an Oracle" baseline uses the same active learning process as Tropel, but instead of crowd responses uses an oracle to respond to the active queries. The oracle responses are obtained using the ground truth labels provided with the CUB dataset.

Table 2 compares the active learning baseline, the linear SVM with all available labels baseline, the Tropel detectors, and an active learning baseline where responses were random. From Table 2, we can see that the crowd creates detectors that are on average slightly better than the oracle baseline. This is likely due to human workers selecting rotated, occluded or otherwise varied examples of the initial visual concept that are missed by only relying on the pre-existing oracle responses. Overall, the detectors trained with all of the labeled data do significantly better. This is due to the fact that many bird species have wide appearance variety that is difficult to capture with only one example. The next section further explores this apparent shortcoming of Tropel.

Hierarchical Similarity

One cause of detector failure on the CUB data set is classifier drift. As observed by Welinder et al. (Welinder et al. 2010), a non-expert crowd may have an incorrect preconceived notion of an attribute or object. In particular, Welinder et al. (Welinder et al. 2010) showed that AMT workers confused grebes for ducks. Figure 6 shows that while a similar error is occurring with our detectors, the detection errors occur in a reasonable way. The top detections of 'Song Sparrow' and 'Caspian Tern' shown in Fig. 6 include the heads of other birds in the Sparrow and Tern families. The top detections for these two classifiers are failing in a predictable way given the taxonomy of this problem.

Table 3 characterizes the detector drift across all bird head

Figure 6: *Classifier Drift*. The two groups of images above show the seed examples and 20 most confident detections of the classifiers trained for the heads of the 'Song Sparrow' and 'Caspian Tern'. These classifiers detect a coherent visual event, but that event is not the same as what is illustrated by the seed examples. The different sparrows have slightly different feathering patterns and the terns have different beak coloration for example.

detectors. In the CUB 200 dataset there are 35 families of birds, such as *Sparrows, Terns, Wrens, Cuckoos*, etc. All species have family membership. Table 3 shows that for several families, the precision of the detector in the 30 most confident test detections for the family is much higher than the precision for a particular species' head detector. The detectors are generalizing to a family of bird heads in the same way we would expect a non-expert human to do. For these particularly fine-grained recognition tasks it is likely that some domain specific instruction would need to be provided to the crowd (e.g. which key features distinguish the Sparrow of interest). The results shown in Table 3 are encouraging for applications such as image retrieval where it is important that the most confident results are all or mostly correct.

Worker Consensus Protocol Comparison

The Tropel system is intended for use on entirely unlabeled datasets. Thus in a typical use case we do have ground truth that is available for the CUB dataset (which we have used to evaluate the detectors).

When creating detectors with the CUB dataset, we are able to use the gold standard positives to score the accuracy of our crowd workers. Our embedded gold standard tests include 5 ground truth positives, all ground truth head patches for the given species, and 5 negatives, which were randomly selected negative patches from the CUB dataset. The test image patches are randomly interspersed with the active query images on every human intelligence task (HIT).

We found that assessing worker recall (true positives/(true positives + false negatives)) is a more informative metric for worker behavior than precision. Selecting only correct examples was easy for workers but finding all of the positive catch trails was hard. The catch trial recall was an average of 0.318, and [0.264, 0.338, 0.395, 0.517] across the [25%, 50%, 75%, and 100%] quartiles. For comparison, the recall of the authors of this paper over 25 HITs was 0.462.

Some workers are good, some mediocre, and some apathetic in that they never identify any positive training examples. It would be helpful if we could even approximately determine which workers to preferentially trust. Tropel seeks

	Precision in Top 30 Detections	
Bird Species	**Species**	**Family**
Cardinal	0.93	0.93
Green Violetear	0.80	0.87
Horned Puffin	0.77	0.80
European Goldfinch	0.70	0.73
American Goldfinch	0.67	0.67
Cape Glossy Starling	0.60	1.00
Scarlet Tanager	0.60	0.93
Purple Finch	0.57	0.80
Song Sparrow	0.30	0.60
Le Conte Sparrow	0.17	0.80
Tree Sparrow	0.17	0.63
Clay colored Sparrow	0.17	0.63
Field Sparrow	0.27	0.53
House Sparrow	0.13	0.23
White throated Sparrow	0.03	0.77
Lincoln Sparrow	0.03	0.90
Western Gull	0.40	1.00
Ivory Gull	0.43	0.80
Herring Gull	0.17	1.00
California Gull	0.10	1.00
Ring billed Gull	0.10	0.63
Heermann Gull	0.27	0.27
Glaucous winged Gull	0.10	0.83
Slaty backed Gull	0.03	0.60
Artic Tern	0.53	0.90
Elegant Tern	0.13	0.97
Caspian Tern	0.20	0.97
Forsters Tern	0.13	0.97
Least Tern	0.07	0.43
Common Tern	0.10	0.33
Prairie Warbler	0.47	0.90
Bay breasted Warbler	0.33	0.33
Prothonotary Warbler	0.43	0.97
Black and white Warbler	0.33	0.37
Golden winged Warbler	0.23	0.63
Canada Warbler	0.10	0.70
Kentucky Warbler	0.50	0.93
Yellow Warbler	0.27	0.80
Pine Warbler	0.23	0.87
Cape May Warbler	0.03	0.77
Black throated Blue Warbler	0.03	0.13
Mourning Warbler	0.07	0.30

Table 3: *Hierarchical Similarity of Top Detections*.The precision scores listed above show the ability of some seemingly low performance detectors to do well at recognizing visually similar bird heads. Precision was calculated as the number of true positives among the 30 most confident detections on the CUB test set. Please note that for each species of bird there are approximately 30 true positive examples in the test set. For bird families such as the *Gulls, Sparrows, Terns,* and *Warblers*, the precision of the individual species detectors are low because those detectors are picking up other, similar looking members of the families. This is shown by the higher family precision scores. This table demonstrates that when our detectors fail, they do so by drifting to detecting visually and semantically similar phenomena.

to avoid the added burden of collecting extra data to generate catch trials. Successful existing alternatives, such as Welinder et al. and Ipeirotis et al., require an extended interaction with a crowd worker to fit parameters to accurately model the workers' expertise and biases (Welinder et al. 2010; Ipeirotis, Provost, and Wang 2010). The simple worker consensus approaches investigated in this paper will not lead to dramatic improvements. However, in keeping with the low-overhead goals of Tropel, we restrict ourselves to work with the relatively weak metadata collected through our UI without any additional instrumentation or data collection.

We examine 3 simple mechanisms which all perform a better than simple consensus. Our workers are ephemeral, and we typically know limited data about them. Across all experiments in this paper so far, the average number of unique workers that participated in creating each detector was 11, out of a possible 12 (4 active queries, with 3 respondents each).

The data easily available to us includes the amount of time a worker spent answering an active query, how many times that worker has worked for us before, and a self-reported confidence value. In the UI shown in Fig. 2, there is a slider bar at the top of the page. Workers are asked to identify their level of confidence about their responses by selecting a value between 1 (low) - 5 (high).

To examine how to estimate a worker's reliability using only these metrics, we created detectors for 20 randomly selected bird types. For these experiments we ask 9 workers (instead of 3) to respond to each active query. The workers votes are weighted by each of the three factors or not weighted at all (the average consensus method). For a patch to be counted as having a true or false label, the margin between the weighted true or false votes has to be greater than 1/3 of the total of the weighted votes. Patches that land inside that margin are discarded.

Table 4 shows that, averaged over the 20 different classifiers, the worker's history with our system was the most reliable metric. However, the improvement in average AP score over the 20 test case classifiers is small compared to using a simple average consensus. Considering how long a worker spent on a task and weighting by the worker's self-reported confidence both outperform average consensus. Of the three metrics we used, the self-reported confidence had the strongest correlation with the workers' own recall, although the correlation coefficient was still a relatively weak 0.19.

Weighting Strategy	**Average AP Score**
Worker Self-Reported Confidence	0.0898
Time Spent Answering Query	0.0874
Number of Previous Tasks Completed	0.1105
Average Consensus	0.0844

Table 4: *Weighted Response Strategies*. The averaged AP scores for the four response weighting methods. Averages are taken over 20 randomly selected bird head detectors.

Across all experiments, 1781 workers participated. Workers were from 33 countries, although 1031 were from the United States and 338 from India. On average, workers where paid approximately $2 per hour.

Cost Comparison

A primary motivation of our pipeline is that it requires relatively few (expensive) trustworthy annotations compared to traditional classifier construction. For example, the CUB dataset has full ground truth annotations and we estimate the cost of training a classifier for a single item on the CUB training set as approximately $60. We estimate the cost of labeling the CUB training set for one part as the number of training images (5994), divided by the number of images shown in each labeling task (25), multiplied by the number of repeat annotators (5) and the fee for each task ($0.05) (Wah et al. 2011). Our estimate does not include the cost of validating annotations and correcting errors in the CUB dataset, tasks which were completed by experts in ornithology and trained volunteers.

The cost of training a detector with our system is $0.60, which is the number of rounds of active queries (4), multiplied by the cost of each query ($0.05) and the number of repeat annotators (3). Note that the cost of training a classifier with our system does not depend on the number of images in the training set, although a larger training set will increase computational training time. Tropel is more computationally expensive than the baseline method, taking an average of 260 cpu hours to train one detector. Training a single linear SVM on the CUB training set takes approximately 10 mins. on comparable hardware. The calculation steps in Tropel are embarrassingly parallelizable however.

Experimental Evaluation - Fashion dataset

Using our crowd-powered Tropel system, we are able to train detectors on a fresh, unlabeled dataset. To test our pipeline with unlabeled data, we obtain a set of unlabeled images from theSartorialist.com, which is a website devoted to images of contemporary fashion worn on the street in New York, London, Milan, Tokyo and Paris. We selected this set of images because they are all taken by a single professional photographer, thus their quality is high. The environment and pose of the subjects are complex and unpredictable. All of the images are taken of people walking the streets of the aforementioned cities. Most pictures contain busy backgrounds and multiple subjects. People are sitting, standing, walking, and interacting with each other without necessarily paying any attention to the photographer. This gives us a challenging dataset with which to test our pipeline. This dataset contains 4785 images. A self-described fashion expert on our team collected the example images for the fashion items using Google image search.

Figure 7 shows the top detections of 10 fashion detectors trained using 5 iterations of our system. In this case study, the dataset was not divided into test and train. Our goal was to create detectors using all available data. This case study is similar to the image retrieval problem in that respect.

As the crowd workers selected positive training examples, those image patches and any patches with an intersection over union (IoU) with the positive patch of 0.5 were removed from the pool of unlabeled images that could be used

in the next active learning iteration. Some image patches that are zoomed out or shifted versions of the worker selected patches are left in the pool of possible training examples. Our goal was to make it possible to get a diverse set of patches that show a given positive example at slightly different perspectives.

Figure 7 shows that the most confident detections are quite similar to the worker selected positives. Where the detectors fail, it is because the dataset didn't contain many examples of the desired item, for example *women's shorts* or *epaulet*.

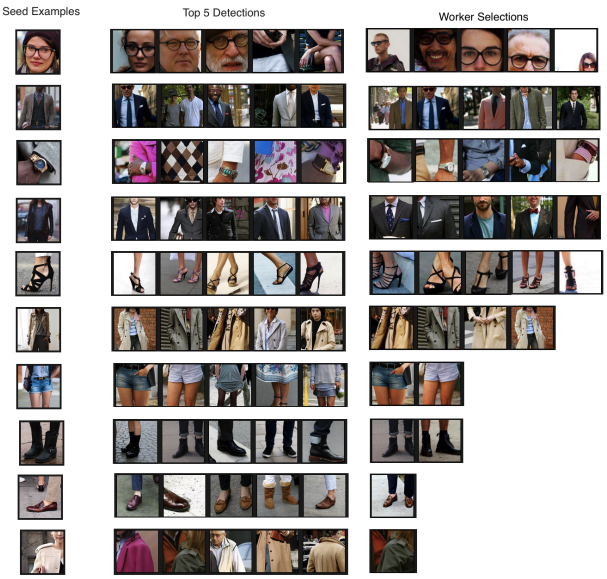

Figure 7: *Example detections from 5 fashion concepts*. This figure shows the 5 most confident detections of on our fashion dataset. Results for the following classifiers are shown: *glasses, men's blazer, watch, jacket, strappy heels, trench coat, women's shorts, boots, men's loafers*, and *epaulet*. On the far right, 5 randomly sampled worker selected positive patches are shown. If the workers selected fewer than 5 positive training examples, all of the positives are shown. The results of the watch and epaulet detectors are especially interesting as those items are physically small, thus making them a much harder detection challenge.

Detecting difficult to name concepts

Another advantageous property of Tropel is that it can train detectors for concepts that aren't easy to name. For instance, what if a user wants to detect architectural elements similar to the Parisian bridge span shown in Fig. 8? There exists no annotated dataset to train such a detector. We use an unlabeled dataset of 70 thousand Paris photos from *flickr.com* for the active learning process. Figure 9 shows the top 5 most confident detections for five distinctive building elements from Paris street scenes. These five examples were identified by Tropel users using the selection tool show in Fig. 8.

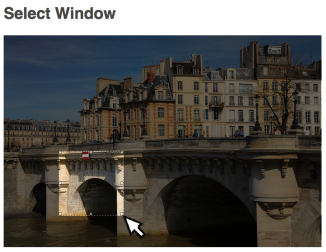

Figure 8: *Seed Patch Selection*. A user selects the region which will seed the crowd active learning of a detector.

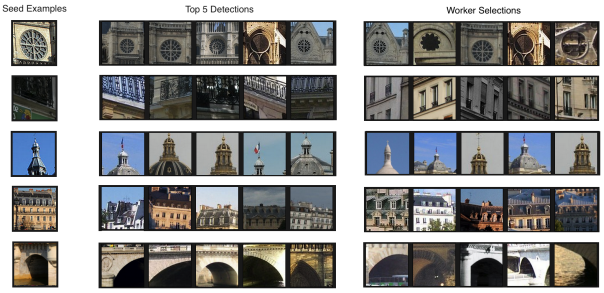

Figure 9: *Example detections for 5 architectural concepts*. While these are distinctive architectural elements (from top to bottom: *Catherine windows, exterior wrought iron balustrades, cupula lanterns, Mansard-roofed apartment buildings*, and *stone bridges*) the end user and the crowd didn't use these names or have architectural expertise.

The detectors shown in Fig. 9 are qualitatively similar to the discriminative patch detectors found automatically by Doersch et al. in "What makes Paris look like Paris?" (Doersch et al. 2012) by mining a collection of Paris and non-Paris photographs. Our scenario is different in that a user directs which concept should be detected.

The pipeline presented in Doersch et al. automatically discovers discriminative architectural details from an unlabeled dataset of street view images from Paris. Our pipeline gives users the opportunity to identify architectural details they find discriminative or important, and directly create classifiers for those often unnamable things. To the authors' knowledge, active learning has not been employed in the literature to create classifiers for non-nameable visual events. Figure 8 shows how a user could select a tricky to describe visual element to seed the detector creation process.

Conclusion

We have shown that the crowd is surprisingly capable of training detectors for specific, fine-grained visual phenomena. An end user can spend minimal effort to provide one or five training examples which, by themselves, are not capable of generating an accurate detector. The crowd can learn from these few examples to build a high precision detector. Our investigations into the hierarchical similarity of the Tropel detectors' output showed how fine-grained a detector it is possible to obtain with limited instructions. The more

limited the instructions to the crowd, the more the crowd generalizes to high level concepts.

Tropel is biased towards high-precision rather than complete recall. It may be possible to change the active query strategy to achieve better recall. In this paper, we opted not to investigate alternative active query strategies to limit the scope of our experimentation. However, improving recall is an important next step.

The linear SVM classifier and CNN image features selected for these experiments likely underperform domain specific detection strategies and fine-tuned deep networks. Still, our active learning system can create successful detectors in three disparate visual domains. Our work shows that it is possible to bootstrap classifiers from a single visual example that approach the performance of traditional baselines yet are still inexpensive in both time and capital investment.

Acknowledgements. The authors would like to thanks Steve Branson and Vansh Kumar at Caltech and Wonnie Sim at Brown for their support for these experiments. Genevieve Patterson is supported by the Department of Defense (DoD) through the National Defense Science & Engineering Graduate Fellowship (NDSEG) Program. Additional support provided by Google Focused Research Award to Serge Belongie and Pietro Perona. James Hays is supported by NSF CAREER Award 1149853.

References

Abramson, Y., and Freund, Y. 2004. Active learning for visual object recognition. Technical report, Technical report, UCSD.

Chen, X.; Shrivastava, A.; and Gupta, A. 2013. NEIL: Extracting Visual Knowledge from Web Data. In *International Conference on Computer Vision (ICCV)*.

Chilton, L. B.; Little, G.; Edge, D.; Weld, D. S.; and Landay, J. A. 2013. Cascade: crowdsourcing taxonomy creation. In *Proceedings of the 2013 ACM annual conference on Human factors in computing systems*, 1999–2008. ACM.

Collins, B.; Deng, J.; Li, K.; and Fei-Fei, L. 2008. Towards scalable dataset construction: An active learning approach. In *Computer Vision–ECCV 2008*. Springer. 86–98.

Deng, J.; Li, K.; Do, M.; Su, H.; and Fei-Fei, L. 2009. Construction and Analysis of a Large Scale Image Ontology. Vision Sciences Society.

Deng, J.; Krause, J.; and Fei-Fei, L. 2013. Fine-grained crowdsourcing for fine-grained recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Divvala, S.; Farhadi, A.; and Guestrin, C. 2014. Learning everything about anything: Webly-supervised visual concept learning.

Doersch, C.; Singh, S.; Gupta, A.; Sivic, J.; and Efros, A. A. 2012. What makes paris look like paris? *ACM Transactions on Graphics (TOG)* 31(4):101.

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

Dow, S.; Kulkarni, A.; Klemmer, S.; and Hartmann, B. 2012. Shepherding the crowd yields better work. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, 1013–1022. ACM.

Everingham, M.; Van Gool, L.; Williams, C. K. I.; Winn, J.; and Zisserman, A. 2010. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision* 88(2):303–338.

Gomes, R.; Welinder, P.; Krause, A.; and Perona, P. 2011. Crowdclustering. In *Neural Information Processing Systems (NIPS)*.

Ipeirotis, P. G.; Provost, F.; and Wang, J. 2010. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*, 64–67. ACM.

Little, G.; Chilton, L. B.; Goldman, M.; and Miller, R. C. 2009. Turkit: tools for iterative tasks on mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*, 29–30. ACM.

Little, G.; Chilton, L. B.; Goldman, M.; and Miller, R. C. 2010. Exploring iterative and parallel human computation processes. In *Proceedings of the ACM SIGKDD workshop on human computation*, 68–76. ACM.

Perona, P. 2010. Vision of a visipedia. *Proceedings of the IEEE* 1526–1534.

Razavian, A. S.; Azizpour, H.; Sullivan, J.; and Carlsson, S. 2014. Cnn features off-the-shelf: an astounding baseline for recognition. *arXiv preprint arXiv:1403.6382*.

Settles, B. 2010. Active learning literature survey. *University of Wisconsin, Madison*.

Vijayanarasimhan, S., and Grauman, K. 2011. Large-scale live active learning: Training object detectors with crawled data and crowds. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, 1449–1456. IEEE.

Wah, C.; Branson, S.; Welinder, P.; Perona, P.; and Belongie, S. 2011. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology.

Welinder, P.; Branson, S.; Perona, P.; and Belongie, S. J. 2010. The multidimensional wisdom of crowds. In *Advances in Neural Information Processing Systems*, 2424–2432.