Human Computation for Object Detection

Rajan Vaish[1], Sascha T. Ishikawa[1], Sheng Lundquist[2], Reid Porter[2], James Davis[1]
University of California at Santa Cruz[1], Los Alamos National Laboratory[2]
{rvaish, stishika, davis}@cs.ucsc.edu, {slundquist, rporter}@lanl.gov

May 09, 2013

TR Number: UCSC-SOE-15-03

Abstract

Object detection is a key component of many computer vision systems. This paper investigates human computation as a platform for object detection. We investigate and report on the quality of raw labels obtained from several sources of human labor. We find that the performance is not the same in all cases. We investigate and report on the quality of a simple aggregation algorithm. Finally, we compare our human computation based object detector against existing computer vision based approaches.

Introduction

Object detection in images is an important component of many existing and possible computer systems. These include faces in photographs, pedestrians in traffic scenes, insects in agricultural monitoring, fish in ocean habitat studies, vehicles in satellite imagery, and subatomic particles in scientific imaging.

Although computer vision works well in some domains, it has lacked robustness in others. This remains an active area of research, typically requiring customization of solutions for each new scenario and object type.

Human computation has been shown to produce image labels with accuracy beyond what is possible by automated methods. However, labels produced by humans are not usable directly because they vary in quality, either because of differences in human opinion, or because the task is too difficult for some workers, or due to intentional provision of incorrect labels.

A variety of algorithms have been proposed for filtering and aggregating raw labels provided by workers in human computation systems. However, these algorithms depend on the characteristics and statistics of the raw data itself. For example, some algorithms assume that most human labels are "correct" and some fraction are "incorrect." In this case filtering out the "incorrect" answers using a verification process is sufficient to produce high quality results. On the other hand some combinations of tasks and human computation platforms are not capable of delivering "correct" results, but rather deliver a set of answers over a continuous distribution of varying "correctness." In these cases an algorithm that attempts to simply verify "correct" answers is not an appropriate solution.

Object detection is an example of a domain that produces a continuous distribution of answers, rather than correct answers. Workers draw bounding boxes that are unlikely to match some notion of ground truth exactly at a pixel level. Instead, the labeled boxes form a distribution in the general area of the object to be located. If the workers are very precise these boxes will be tightly clustered. If the workers are less precise, there might be a large variance in the boxes provided. In either case, an algorithm that aggregates multiple labels into a single answer could provide quality exceeding the best individual label.

In this paper, we investigate and report on the quality of raw labels obtained from several sources of human labor. We find that the quality is not the same in all cases, and thus is an important parameter of the expected performance of any algorithm. In addition, all of our raw labels are relatively inaccurate. The majority of labeled bounding boxes have less than 80% overlap with ground truth.

We investigate a simple algorithm to aggregate the raw labels. We follow other researchers by first eliminating outliers. Next, since our test images have multiple objects to label, we cluster the raw bounding boxes into sets, each of which is aggregated as the mean value of available labels. The result is quality above what is obtained from individual workers.

Our datasets were chosen from the computer vision literature, and we are thus able to compare directly, the quality obtained through human computation, to the quality obtained via computer vision methods. Since the published computer vision accuracy curves are the result of careful domain specific optimization, we also report the quality

obtained by implementing a "standard" object detector. The human-produced labels processed by our algorithm greatly exceed the accuracy available from the published computer vision algorithms on these datasets.

The main contribution of this work is providing an analysis of the object detection task on human computation platforms. We investigate the characteristics of raw labels on more than one platform, the effects of aggregation, and the improvements available versus existing computer vision solutions.

Related Work

A rapidly growing number of large-scale, human-powered data collection platforms have been built (Little, J. 2012; Deng, J. 2009), many of which leverage Amazon's Mechanical Turk for human labor. Collecting image annotations and labels is one of many applications of crowdsourcing and bounding boxes (Welinder, P. 2010; Whitehill, J. 2009) are a popular annotation method. Other techniques such as polygons and scissor tools have also been explored (Sorokin, A. 2008; Little, J. 2012).

All of these papers use raw annotations from workers and verification processes to exclude poor results. Our work analyzes a method for aggregating the raw results from multiple untrained crowd workers, showing that it improves the final annotation quality.

High-quality bounding boxes have been explicitly investigated using human computation tasks for quality and coverage verification. For example, a study by Su et al. obtained excellent results and reported that 62% of raw worker boxes are "visibly tight" (Su et al., 2012). However, our observed worker quality shows much lower accuracy, and it is not clear that a verification-based algorithm would be appropriate under our observed conditions.

Efforts have been made to combine machine learning or computer vision with crowdsourcing to optimize the flow of task requests (Quinn, A. 2010; Vittayakorn, S. 2011; Branson, S. 2011; Vijayanarasimhan, S. 2011). Similarly, explicit comparisons have been conducted comparing CPU with human cost, and a balance between the usages of two has been suggested (Vondrick, C. 2010). This work is orthogonal to the goal of our work, to aggregate labels once they are obtained.

Aggregation of weak classifiers has been studied extensively in machine learning and machine vision (Tao, L. 2007). Weak classifiers of higher-level data such as image segmentations have also been combined (Ghosh, S. 2009). Approximate variational methods, including belief propagation and mean field have been applied to crowdsourced labels (Liu, Q., 2012). However, none of these studies explicitly address the aggregation of bounding box labels obtained from crowdsourced workers.

Object detection has been extensively studied in computer vision on images of all types, quality and content (Lempitsky, V. 2009; Dollar, P. 2009). The data in our work consists of images of pedestrian (Leibe, B. 2007) and bumblebees (Miranda, B. 2012) taken from existing computer vision research. We show that crowdsourced object detection performs substantially better than CPU-only methods.

Experimental Setup

Datasets

In our study we use two datasets, each of which was drawn from the existing computer vision literature on object detection.

Bumblebee: Miranda et al. studied the behavior of bumblebees while pollinating flowers (Miranda, B. 2012). In our study we used 411 (of a total of 3,237 frames) 640x480 pixel JPEG images. A camera was pointed towards a sweet pepper flower that was nearly stationary. The camera records the arrival and departure of a single bumblebee that spends most of its time on the flower. Furthermore, the background is static and there is no occlusion. One frame from this dataset is shown in Figure 1(a).

(a) *(b)*

Figure 1: Example frames from the datasets used to investigate object detection. Note that the bee in (a) is blurry and thus hard to localize precisely for both human workers and automated algorithms. The pedestrians (b) are sometimes partially occluded, making the problem challenging.

Pedestrian: Leibe et al. studied multi-object detection and tracking in the context of pedestrians (Leibe, B. 2007). We used 76 frames of their dataset, consisting of 320x240 images with noticeable compression artifacts. The frames contain multiple pedestrians, often with heavy occlusion, and a dynamic background. One frame from this dataset is shown in Figure 1(b).

Worker Interface

The images were annotated using a custom web-based annotation tool shown in Figure 2. This interface contains English language instructions to draw tight bounding

boxes, as well as simple visual examples. The main work panel allows drawing bounding boxes around multiple objects, and contains a submit button. Images are displayed in random order, and workers annotate multiple images as part of a single task. The interface is implemented in Processing, and the web backend is implemented with PHP and Javascript/Ajax code that records all entries in a MySQL database.

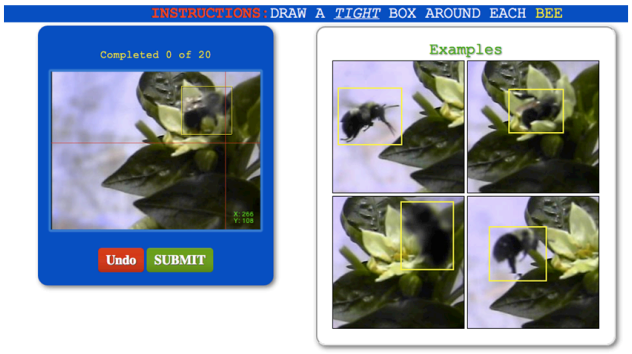


Figure 2: User interface where human workers were asked to draw bounding boxes around objects of interest. The interface allows drawing multiple boxes when there are multiple objects to detect. Examples of correct boxes are shown next to the work area as a reference to workers.

Aggregation Methods

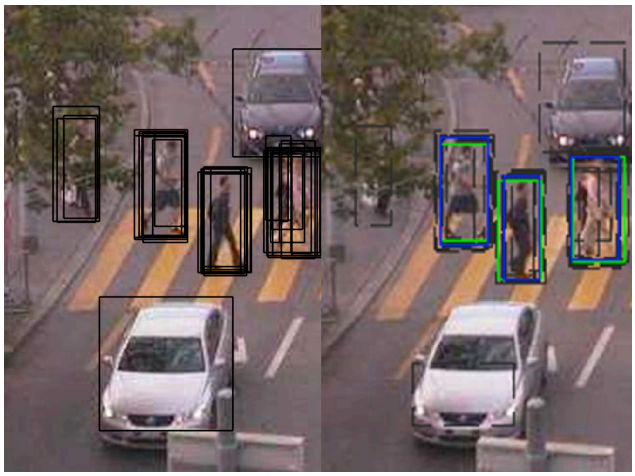
Multiple workers provide a set of bounding boxes, which vary greatly in quality. Figure 3(a) shows an example of all boxes received on one query.

One of the primary challenges of aggregation is determining the correspondence between bounding boxes and the objects they inscribe, especially when multiple instances of an object are visible in an image. For each image we use a mean-shift clustering algorithm on the entire set of labels to assign boxes from different workers to clusters. Each cluster represents a single instance of the object and is aggregated separately.

A secondary challenge is to remove outlier boxes due to mistakes or intentional mislabeling so that they do not adversely affect the final answers. For example, in Figure 3(a) worker(s) incorrectly labeled cars. We implemented outlier rejection based on worker consensus and bounding box size.

Mean-shift clustering algorithm

The mean-shift algorithm (Cheng, Y. 1995; Comaniciu, D. 2002) is a nonparametric clustering technique that does not require the number of clusters to be specified in advance. Mean-shift treats the points in an *N*-dimensional feature space as a probability density where dense regions correspond to local maxima – or modes – of the underlying distribution. These local maxima are obtained by randomly assigning circular search windows of a specified size, or bandwidth, within the feature space. A gradient ascent algorithm is used to find the approximate locations of the modes of each distribution. Finally, points that are closest to each mode are assigned as members of each cluster.



(a) *(b)*

Figure 3: (a) Bounding boxes provided by several workers are shown. Although most workers correctly labeled pedestrians, some incorrectly labeled cars. (b) Blue boxes are the result of the mean shift clustering algorithm followed by outlier rejection and aggregation. Green boxes are ground truth labels. Note that the clusters of raw bounding boxes were replaced by a single aggregate bounding box for each pedestrian.

Bounding boxes within a single cluster are aggregated by taking the mean of the dimensions of the boxes within the cluster. The results of this algorithm are shown in Figure 3(b). Blue boxes are the result of our clustering algorithm. Note that groups of individual labels from workers have been correctly clustered into a single bounding box around each pedestrian. Green boxes are ground truth labels provided with the dataset.

Consensus voting outlier rejection

The degree of consensus among workers is one metric that can be used to remove outliers. Wrong labels, such as the boxes around cars in Figure 3(a) can be detected because few workers provide these labels.

We define consensus outlier rejection as follows. Given an image, we obtain the corresponding set of N bounding boxes from the set $\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_N\}$ of all the raw estimates of object locations created by human workers. Each bounding box $\mathbf{b}_k \in \mathbf{B}$ is specified by the vector $\langle x, y, w, h \rangle$, where x and y represent the pixel coordinates of the top left corner of the box and w and h represent the width and

height of the box, respectively. For each $\mathbf{b}_k \in \mathbf{B}$ we create a corresponding binary mask $\mathbf{m}_k$, the same dimensions as the image, in which all pixels inside $\mathbf{b}_k$ are 1 (white) and those outside of it are 0 (black). By summing the individual masks we obtain the consensus map $\mathbf{M}$ (Figure 4), which represents the distribution of total pixel-wise votes across the image. Given the consensus map of an image, the average consensus score s_k for some bounding box $\mathbf{b}_k$ is computed below.To reject consensus outliers we remove all bounding boxes with scores less than a threshold value C.

$$s_k = \frac{1}{w_k h_k} \sum_{j=y}^{y+h} \sum_{i=x}^{x+w} \mathbf{M}(i,j)$$

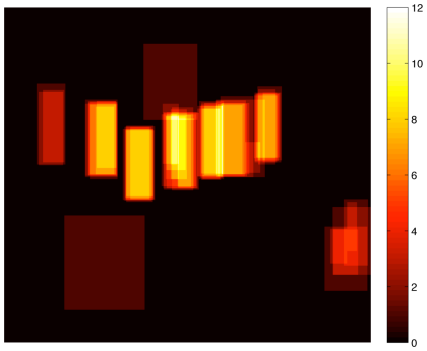


Figure 4: Consensus map of the agreement between workers is shown. Note that incorrect boxes drawn around cars have comparatively low consensus whereas locations with pedestrians have been labeled by more workers. We reject bounding boxes in areas of low consensus.

Size-based outlier rejection

We noticed that very small bounding boxes, probably created by accident, sometimes fell entirely within a high consensus area. We also observed larger bounding boxes that contain multiple pedestrians. Although in both cases they are in the right general region of an actual object, they do not represent the actual size dimensions of a correct object and adversely affect the clustering algorithm.

To remove these boxes we used size based outlier rejection. Given the set of all bounding boxes $\{\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_N\}$, any box with width or height more than 3 standard deviations from the mean was removed prior to clustering.

Experimental Findings

In this section, we'll discuss our experimental findings:

Worker Demographics

Since the applicability and robustness of our algorithm is related to the performance of workers, we investigated multiple crowdsourcing platforms, each employing different workers. We tried two structured for-pay crowdsourcing platforms, Amazon's Mechanical Turk and an upcoming alternative that we will refer to simply as Microtask-X (since it is a much smaller and newer company whose performance is likely to change).

We also investigated an unstructured method of simply asking all our Facebook friends to participate. In all cases the workers were directed to an identical work environment hosted on our own server. Workers on Mechanical Turk were paid $0.15 per set of 5 images. Microtask-X set a pay rate outside our control, charging us $0.05 per set of 20 images. On Facebook, the volunteers were unpaid.

The majority of Mechanical Turk workers were from India (82% and 64% of workers for the pedestrian and bumblebee tasks, respectively) in all trials, with only a small fraction from each of several countries including USA, Canada, and UK. In contrast, Microtask-X had a more diverse workforce that varied across trials with the majority of workers from Philippines and USA (39% and 28%, respectively) in the pedestrian trials. Most workers in the bumblebee trials were from Pakistan, Philippines, and India (32%, 24%, 20%, respectively).

Latency

The time workers take to complete tasks is an important dimension in human computation algorithms since it is related to both latency and the necessary pay rate to engage workers.

	Bumblebee		Pedestrian	
Platform	**Median**	**Mean**	**Median**	**Mean**
AMT	11	19.8	68	87.4
MTX	9	13.3	58	210.1
FB	8	22.2	-	-

Figure 5: Time taken per image for workers to label bounding boxes.

Figure 5 reports the time workers took on individual images in seconds. Notice that the completion time is roughly similar across services. Because some workers take a very long time, the distribution is heavy tailed and the mean is much greater than the median. The pedestrian tasks take longer per image because there are multiple smaller bounding boxes to label as compared to the one large box in the bumblebee dataset. We next looked at the overall rate at which labels were obtained. This might not be correlated with the time taken on individual tasks since workers could simply not be engaged on the project.

We compare the rate of annotation acquisition for our Bumblebee dataset on Amazon Mechanical Turk (AMT), Microtask-X (MTX), and by asking our Facebook (FB)

friends in Figure 6. In this case MTX provided labels much more quickly than AMT. As might be expected, attempting to ask our friends for free labor in the name of research resulted in a much slower rate of acquisition. This rate was sufficiently slow that we ceased using Facebook on subsequent experiments.

Our comparison of the AMT and MTX platforms for the pedestrian dataset (not shown) acquired labels much faster since the pedestrian dataset results in multiple annotations per image.

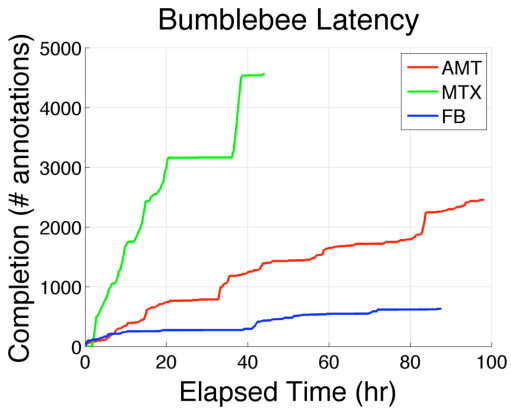


Figure 6: Cumulative number of responses since the job submission for the bumblebee dataset. In this case the AMT microtask platform provided results much faster than MTX. Attempting to ask Facebook (FB) friends to help with our research project resulted in a slow rate of response.

Quality of work

The algorithms appropriate for aggregating and processing crowdsourced data are a function of the characteristics of the raw data. In addition, the accuracy of work might vary between platforms. We thus investigated the quality of raw bounding box labels.

The existing literature defines "accurate" in a variety of ways. One study on crowdsourcing bounding boxes, labeled as "accurate" only bounding boxes that were a very tight fit to the object, overlapping nearly 100% (Su, H. 2012). In contrast, another study using computer vision defined any bounding box that overlapped with ground truth by more than 50% as "accurate" (Leibe, B. 2007). We report the quality of bounding boxes generated by participants using the intersection-over-union metric presented in (Everingham, M., et al. 2005). A fitness score between two bounding boxes is computed as the quotient of the overlapping area to the area spanned by the union of both boxes. Rather than choose an arbitrary threshold as "accurate," we report the distribution of observed qualities.

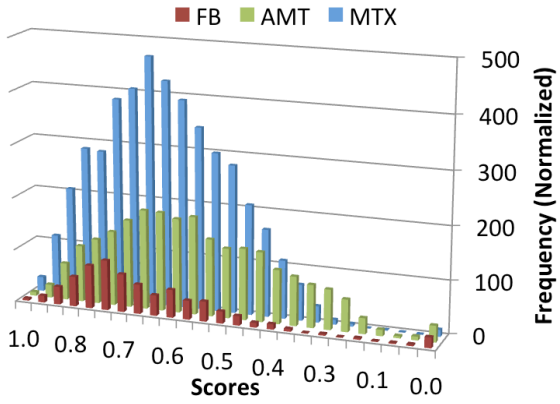


Figure 7: Raw score distribution for the bumblebee dataset varies greatly across services. Nearly all of the labels accumulated from Facebook friends had at least 40% overlap with ground truth, while a large fraction of AMT labels fell below this quality threshold.

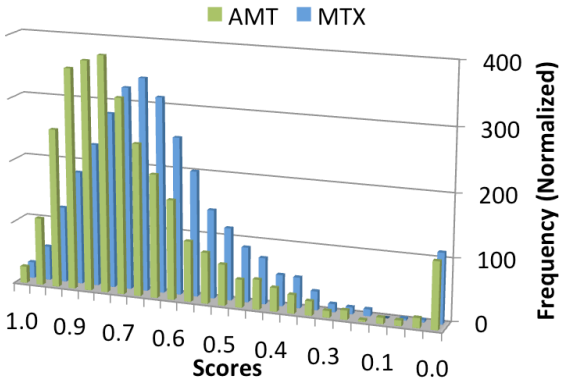


Figure 8: Quality distribution on the pedestrian dataset was noticeably higher, and in this case AMT performed comparatively well.

The distribution of quality scores obtained from each crowdsourcing platform on the bumblebee dataset is shown in Figure 7. The horizontal axis reports scores in the range [0,1], where 0 indicates the bounding box did not overlap with any ground truth and 1 indicates a perfect overlap with ground truth. The vertical axis represents the total number of annotations collected in the dataset. Note that the distributions of raw labels are not identical. Nearly all Facebook labels had at least a 40% overlap with ground truth, while a substantial number of Mechanical Turk labels fell below this level. Differences can also be seen in the percentage of labels that are completely wrong (0% overlap to ground truth). On paid microtask services this likely corresponds to intentional spammers, and rejecting these answers is a component of many human computation

algorithms. In this case, Mechanical Turk had a larger number of responses in this category than did Microtask-X.

The results from our experiments with the pedestrian dataset are shown in Figure 8. In this case AMT had a higher distribution of quality than MTX.

Effect of algorithm parameters

Our aggregation method has parameters in both the mean-shift clustering and consensus voting stages. These can be used to adjust the relationship between the number of true and false positives. Following the convention of Everingham et al. we define true positives as detected bounding boxes that overlap with a ground truth box by at least 50% (Everingham, M., et al. 2005). Boxes which do not sufficiently overlap with ground truth boxes are labeled false positives. If two boxes from our algorithm overlap with the same ground truth box, the second is labeled as a false positive.

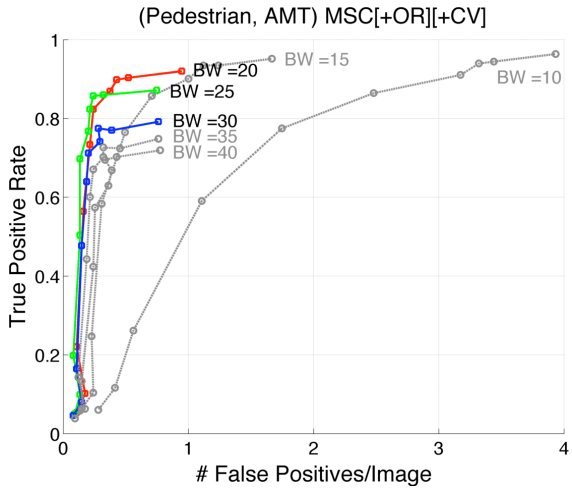


Figure 9: Our aggregation algorithm was evaluated across a range of parameters. Lines are plotted for fixed values of clustering bandwidth (BW) with the consensus voting threshold (C) varied.

The results of running our algorithm with different parameter settings on the pedestrian dataset are shown in Figure 9. Lines are plotted at different values of the mean shift clustering bandwidth (BW). Along each line the consensus voting threshold (C) is varied. When C is 1 a large number of false positives occur since no boxes are rejected. As C increases, both the true and false positives are reduced since more boxes are rejected. From the data, we noticed that the bandwidth (BW) parameter is best set at roughly twice size as the bounding boxes. This makes sense because the clustering bandwidth corresponds to the dimensions of the spatial search radius in the image.

The results on the bumblebee dataset are shown in Figure 10. In this case we have plotted lines with different values of the consensus threshold, C. As the clustering bandwidth decreases more clusters are created and both true and false positives increase. From the data we see that the consensus voting threshold is best set well above 1, but several votes less than the total number of voters available.

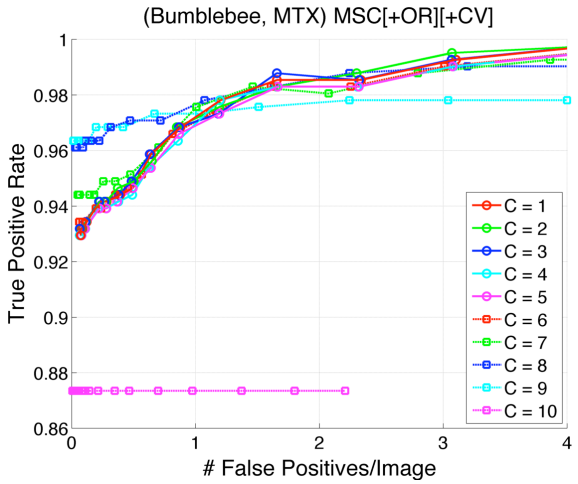


Figure 10: The bumblebee dataset is shown, this time evaluated by holding the consensus voting threshold (C) constant and varying the clustering bandwidth (BW). Note that the algorithm behaves well over a large range of parameters. The y-axis scale is set to a very high range to allow visibility.

Comparison with computer vision

Object detection is traditionally framed as a computer vision problem, so we compare our results against those from published computer vision research. We find that even our simple human computation based detector performs substantially better than existing computer vision solutions.

Pedestrians: The pedestrian dataset has been the target of several prior studies. A fixed viewpoint object detector operating only on single frames is the closest comparison to the data available to our algorithm (Leibe, B. 2005). There has also been work on building a coupled detector and tracker which makes use of temporal information across frames to improve object localization (Leibe, B. 2007). This later work uses more data than our workers have available, since we present them only individual frames. We follow the error metric of 50% overlap, and reporting methodology of true positive rate versus the number of false positives per image used in both of these prior studies. Since both the computer vision and human computation methods have thresholds to adjust, these are reported as curves. Figure 11 shows our results together with the computer vision results. Human computation provides a much higher detection rate.

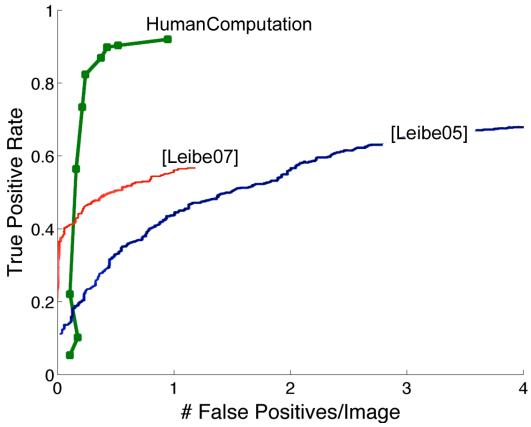


Figure 11: Our human computation algorithm was compared against the results from two prior published computer vision papers on the pedestrian dataset, and found to have a much higher detection rate.

Bumblebee: The bumblebee dataset has been investigated using both a Viola-Jones detector, as well as a detection plus tracking framework that makes use of temporal coherence (Miranda, B. 2012). Prior work on this dataset reports true and false positive *rates*. Given the number of sub-windows scanned per image, these values can be converted into the equivalent metric, false positives per image. The Viola-Jones detector uses between 145,000-575,000 sub-windows to scan each image, depending on parameter settings (Viola, P. 2004). We use the lower number to convert false positive rates into false positives per image.

	True Positive85%	True Positive 96%
[Miranda12] Detection Only	580 false positive/image	
[Miranda12] Detect + Track		145 false positive/image
Human Computation	**0.007** false positive/image	**0.015** false positive/image

Figure 12: The algorithm in this paper is compared against the published accuracy using a computer vision method on the bumblebee dataset. The false positive rate is orders of magnitude lower.

The values from prior work are compared to our method in Figure 12. Miranda et al. give false positive rates in the range of 0.1%, which are equivalent to a high number of false positives per image. Note that our method results in significantly fewer false positives per image at the same true positive rate. Using 411 test images, and at true positive rates chosen to match those given in Miranda et al., our method produced a total of 3 false positives at 85% true positive rate, and 6 false positives at 96% false positive rate. We divide our total false positives by the number of images to get the comparable metric of false positives/image.

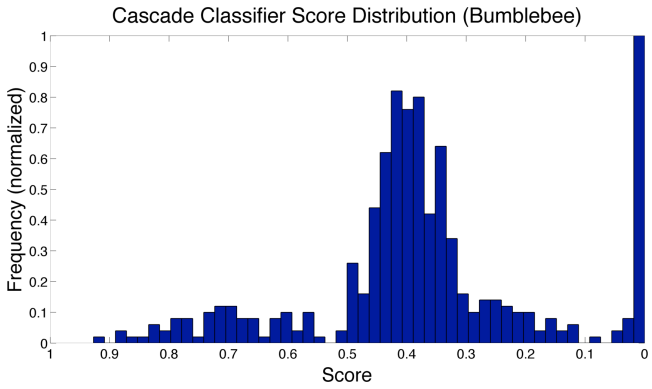


Figure 13: A standard Haar Classifier from OpenCV was used to find bounding boxes in the bumblebee dataset. The quality distribution evaluated on the same (intersection/union) metric is noticeably below any of the human provided label distributions.

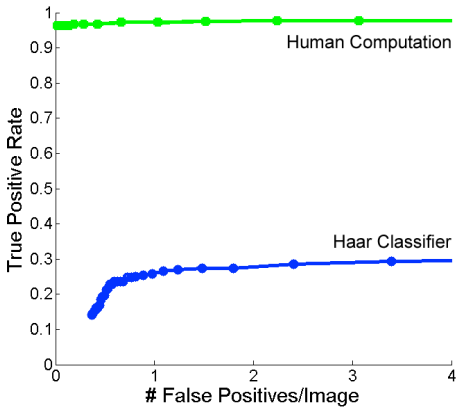


Figure 14: The human computation algorithm in this paper is compared against a standard Haar Classifier as implemented in OpenCV. The standard computer vision solution does not approach the quality of aggregated human labels.

In order to better understand the performance of computer vision based approaches we implemented a standard Haar classifier using OpenCV. This implementation was not heavily tuned to this dataset, so performs worse than published results with custom tuned classifiers. Nevertheless we believe it provides some insight into the performance that can be expected from using a generic unoptimized classifier for object detection.

We trained the classifier with 422 ground truth images. Figure 13 shows the intersection over union scores

obtained. The computer vision algorithm performs noticably worse than the data for human labelers shown earlier.

Lastly we compare the Haar Classifier to our algorithm using an ROC curve similar to that reported in many computer vision papers, shown in Figure 14. OpenCVs detectMultiScale function was used and the minNeighbors parameter was varied. We used the 50% intersection over union metric to determine matches with ground truth. The standard computer vision implementation does not approach the level of quality achieved by human computation.

Conclusions

This paper investigates the use of human computation as a platform for object detection in images.

We presented results showing that different human labor platforms have different accuracy and latency characteristics. We believe that this variability implies a need for designers of human computation algorithms to specify the raw label characteristics on which they expect their algorithms to perform well.

We implement a simple aggregation method to combine results from several workers and investigate its performance. We find that aggregation improves results beyond the quality of individual raw labels.

Finally we compare this human computation based object detector to three different computer vision based object detectors, finding that human computation substantially outperforms computer vision on this task.

Taken together, these findings suggest that micro-task based human labor is a viable platform for developing "computer vision" *algorithms*. Even the extremely simple algorithm investigated in this paper showed improvements in quality. Further, despite the cost associated with human labor, the quality gains were shown to be large enough that deployment could be considered in domains which place a high value on high accuracy.

References

Branson, S.; Perona, P.; and Belongie, S. 2011. Strong supervision from weak annotation: Interactive training of deformable part models. In *ICCV,* 1832–1839.

Comaniciu, D., Meer, P. 2002. Mean shift: A robust approach toward feature Space Analysis. In *the IEEE Transactions on Pattern Analysis and Machine Intelligence,* Volume: 24, Issue: 5, 603 - 619.

Cheng, Y. 1995. Mean shift, mode seeking, and clustering. In *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 17.8 (1995): 790-799.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR'09.*

Dollar, P., Wojek, C., Schiele, B., Perona, P. 2009. Pedestrian Detection: A Benchmark. In *CVPR'09.*

Everingham, M., et al. 2005. The 2005 PASCAL Visual Object Classes Challenge. In *PASCAL Machine Learning Challenges Workshop, MLCW.*

Ghosh, S., Pfeiffer, J.J., Mulligan, J. 2009. A General Framework for Reconciling Multiple Weak Segmentations of an Image. In *WACV,* page 1 - 9. IEEE'09.

Leibe, B., Schindler, K., Van Gool, L. 2007. Coupled Detection and Trajectory Estimation for Multi-Object Tracking. In *ICCV'07.*

Leibe, B., Seemann, E., Schiele, B. 2005. Pedestrian detection in crowded scenes. In *CVPR'05*.

Lempitsky, V., Kohli, P., Rother, C., Sharp, T. 2009. Image Segmentation with A Bounding Box Prior. In *ICCV09.*

Little, J., Abrams, A., Pless, R. 2012. Tools for Richer Crowd Source Image Annotations. In *WACV, page* 369-374. IEEE'12.

Liu, Q., Peng, J., Ihler, A., 2012. Variational Inference for Crowdsourcing. In *NIPS 2012.*

Miranda, B., Salas, J., Vera, P. 2012. Bumblebee Detection and Tracking. *IEEE Workshop on Visual observation and analysis of animal and insect behavior at ICPR, 2012.*

Quinn, A., Bederson, B., Yeh, T., Lin, J,. 2010. CrowdFlow: Integrating Machine Learning with Mechanical Turk for Speed-Cost-Quality Flexibility. In Technical Report HCIL-2010-09, University of Maryland.

Sorokin, A., and Forsyth, D. 2008. Utility data annotation with amazon mechanical turk. *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops* 51(c):1–8.

Su, Hao., Deng, J., Fei-Fei, L. 2012. Crowdsourcing Annotations for Visual Object Detection. In *AAAI Human Computation Workshop, 2012.*

Tao, L., Ding, C., Jordan, M. 2007. Solving Consensus and Semi-supervised Clustering Problems Using Nonnegative Matrix Factorization. In *IEEE ICDM '07*.

Vijayanarasimhan, S., and Grauman, K. 2011. Large-scale live active learning: Training object detectors with crawled data and crowds. In *CVPR,* 1449–1456.

Viola, P., Jones., M. 2004. Robust real-time face detection. *In International journal of computer vision* 57.2 (2004): 137-154.

Vittayakorn, S., and Hays, J. 2011. Quality assessment for crownsourced object annotations. In *Proceeding of British Machine Vision Conference (BMVC).*

Vondrick, C., Ramanan, D., Patterson, D. 2010. Efficiently Scaling Up Video Annotation with Crowdsourced Marketplaces. In *ECCV'10.*

Welinder, P., Perona. P. 2010. Online crowdsourcing: rating annotators and obtaining cost-effective labels. In *CVPR'10.*

Whitehill, J.; Ruvolo, P.;Wu, T.; Bergsma, J.; and Movellan, J. R. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. *In NIPS,* 2035–2043.